2 Data Formats

This chapter provides information about data files that are read by or saved from EnSight.

2.1 Dataset Basics describes the data formats that can be read directly into EnSight. Direct read capability for the files produced by other analysis codes is being developed. If the results from your analysis code is currently not supported, please contact CEI to inquire on availability.

2.2 Reading and Loading Data describes the way in which data is read by and loaded into EnSight and provides suggestions on minimizing memory usage.

2.3 Command files provides a description of the files that can be saved for operations such as automatic restarting, macro generation, archiving, hardcopy output, etc.

2.4 Saving and Archiving describes options for saving, printing and archiving a number of different files, from window positions to archiving the entire current state of the program.

2.5 EnSight File Formats describes the format for all file formats EnSight is capable of reading and writing.

Note: *Formats for the various Analysis codes are not described herein.*

2.1 Dataset Basics

EnSight is designed to be an engineering postprocessor, yet its many features can be used in other areas as well. Its native data is defined as general finite elements or curvilinear structured data. EnSight has been used to visualize and animate results from simulations of diesel combustion, cardiovascular flow, petroleum reservoir migration, pollution dispersion, meteorological flow, and from many other disciplines. EnSight has two native data formats (EnSight5 and EnSight6) which are defined so that they can be easily interfaced to your analysis code.

(see Section 2.5, EnSight File Formats)

EnSight reads node and element definitions from the geometry file and groups elements into an entity called a *Part*. A Part is simply a group of nodes and elements (the Part can contain different element types) which all behave the same way within EnSight and share common display attributes (such as color, line width, etc.).

EnSight allows you to read multiple datasets and work with them individually in the same active session. Each data set comprises a new “Case” and is handled by its own Server process.

EnSight also supports data formats for popular engineering simulation codes and generally used data formats.

Formats Used For Both Computational Fluid Dynamics and Structural Mechanics

- The **EnSight6** format supports the following files:

Case	Defines all of the variables, time steps, etc. that completely describe the files which will be used for an EnSight Case.
Geometry	Defines all geometric model Parts in terms of groups of finite elements, or ijk blocks.
Variable	A file for each variable, which contains either scalar or vector information for every node defined in the geometry file (per_node) or for elements of various parts (per_element).
Measure/Particle	Defines discrete Particles in space directly from a simulation or measured information from an experiment. The measured information can be used to compare actual versus simulated results.

- The **EnSight5** format supports the following files:

Geometry	Defines all geometric model Parts in terms of groups of finite elements.
Result	Defines variable names such as Stress, Strain, and Velocity, and indicates what files these are tied to. It also, defines time information if you have a transient data case. This file is optional (and is unnecessary if your geometry is static and you have no results data).

Variable	A file for each variable, which contains either scalar or vector information for every node defined in the geometry file.
Measure/Particle	Defines discrete Particles in space directly from a simulation or measured information from an experiment. The measured information can be used to compare actual versus simulated results.

- **MPGS4** is composed of the following files:

Geometry	Defines all geometric model Parts in a general n-sided polygon format.
Result	Utilizes the EnSight results file format. This file is optional.
Variable	A file for each variable, which contains either scalar or vector information for every node defined in the geometry file.
Measure/Particle	Utilizes the EnSight measured/Particle file.

Formats Generally Used For Computational Fluid Dynamics

- **ESTET** contains the geometry and results information in one file. This is the native binary data format for the ESTET simulation code. The EnSight measured/Particle file can also be used in conjunction with these.
- **FIDAP Neutral** contains the geometry and results in one file. This file is produced by a separate procedure defined in the FIDAP documentation. If the data is time dependent this information is also defined here. The EnSight measured/Particle file can also be used in conjunction with these.
- **FLUENT Universal** contains the geometry and results in one file. However, FLUENT will produce multiple universal files for transient analysis which are accessed through a slightly modified EnSight result file. The EnSight measured/Particle file can also be used in conjunction with these.
- **N3S** is native to the N3S simulation code and is composed of the files:

Geometry	Defines the geometry.
Result	Contains all result information describing variables and the scalar and vector information. This file is required.
Measure/Particle	Utilizes the EnSight measured/Particle files.

- **PLOT3D** is composed of the following files:

Geometry	Defines the geometry. This is known as a GRID file in PLOT3D and FAST. This file is a structured file format with FAST enhancements.
Result	Utilizes a modified EnSight results file format. This file is optional.

Variable This file is a solution file (Q-file) defined in PLOT3D or a function file as defined by FAST. The modified EnSight results file provides access to multiple solution files that are produced by time dependent simulations.

Measure/Particle Utilizes the EnSight measured/Particle files.

- **STAR-CD** is composed of the following files:

  Geometry Defines the geometry. This is “file16”.

  Result Contains results information. This can be either “file9” for steady state info or “file29” for transient information. This file is optional.

  Measure/Particle Utilizes the EnSight measured/Particle files.

- **FAST UNSTRUCTURED** is composed of the following files:

  Geometry Defines the geometry as unstructured triangles and/or tetrahedrons. It is the FAST unstructured single block grid file.

  Result Utilizes a modified EnSight results file format. This file is optional.

  Variable This file is a solution file (Q-file) defined in PLOT3D or a function file as defined by FAST, with I equal to the number of points and J=K=1. The modified EnSight results file provides access to multiple solution files that are produced by time dependent simulations.

  Measure/Particle Utilizes the EnSight measured/Particle files.

Formats Generally Used For Structural Mechanics

- **ABAQUS** can produce a .fil file which contains the geometry and results requested. This file must be generated in ASCII format in order to be read by EnSight. EnSight will read the commonly used nodal results and the commonly used element based results found in this file. However, the element based results must be averaged at the nodes. Additionally, if the .dat file containing element sets is provided, EnSight will create Parts based on these element sets.
- **ANSYS RESULTS** contains the geometry and results in one file. The files are defined as .rst, .rth, rfl, and .rmg files in the ANSYS documentation (EnSight 5.5 supports only the .rst file). If the data is time dependent this information is also defined here. The EnSight measured/ Particle file can also be used in conjunction with these.

- **Movie.BYU** is composed of the following files:

Geometry	Defines all geometric model Parts in a general n-sided polygon format.
Result	Utilizes the EnSight results file format. This file is optional.
Variable	A file for each variable, which contains either scalar or vector information for every node defined in the geometry file.
Measure/Particle	Utilizes the EnSight measured/Particle files.

Data files are never altered by EnSight. They are used only for reading the dataset information. EnSight can produce a set of files in its native format to save geometric information that may have been read from another format or created through the postprocessing techniques. Section 2.4, Saving and Archiving

2.2 Reading & Loading Data

Reading and then Loading Data into EnSight is a two step process. First, files are specified through the File Selection Dialog and then read by EnSight to the Server. Data from the files is then loaded to the Client using the Data Part Loader dialog. All Parts or a subset of those available on the Server may be loaded to the Client. You should try to reduce the amount of information that is being processed in order to minimize required memory. Here are some suggestions:

- When writing out data from your analysis software, consider what information will actually be required for postprocessing. Any filtering operation you can do at this step greatly reduces the amount of time it takes to perform the postprocessing.
- Load to the Client only those Parts that you need. For example, if you were postprocessing the air flow around an aircraft you would normally not need to see the flow field itself, but you would like to see the aircraft surface and Parts created based on the flow field (which remains available on the Server).
- For each Part you do load to the Client, a *representation* must be chosen. This visual representation can be made very simple (through the use of the Feature Angle option), or can be made complex (by showing all of the surface elements). The more you can reduce the visual representation, the faster the graphics processing will occur on the Client (see Node, Element, and Line Attributes in (see Section 3.3, Part Editing)).
- If you have multiple variables in your result file, activate only those variables you want to work with. When you finish using a variable, consider deactivating it to free up memory and thereby speed processing (see Section 4.1, Variable Selection and Activation).
- When dealing with transient data in an EnSight flipbook, consider loading initially only a sampling of the available time steps—you can always load the in-between steps later if you find something interesting.

Reading EnSight6 Data

EnSight6 input data consists of the following files:

- Case file (required)
- Geometry file (required)
- Variable files (optional)
- Measured/Particle files (optional)

  - Measured/Particle geometry files
  - Measured/Particle results files
  - Measured/Particle variable files

The Case file is a small ASCII file which defines geometry and variable files and names, as well as time information. The Case file points to all other files which pertain to the model. The geometry file is a general finite-element format describing nodes and Parts, each Part being a collection of elements, and/or structured curvilinear ijk blocks. Measured/Particle files contain data about discrete Particles in space from the simulation code or information directly from actual experimental tests.

EnSight6 data is based on Parts. The Parts defined in the data are always available on the Server. However, all Parts do not have to be loaded to the Client for display. Large flow fields for CFD problems, for example, are needed for computation by the Server, but do not generally need to be seen graphically.

EnSight6 data can have changing geometry, in which case the changing geometry file names are contained in the Case file.

File Selection dialog

Figure 2-1

File Selection dialog for EnSight6 data

The File Selection dialog is used to specify which files you wish to read.
`Access: Main Menu > File > Data (Reader) …`

Filter	This field specifies the directory name that your data files reside in. Enter a /* at the end of the name to list all of the files and directories contained there. To filter to a smaller file list you can be more specific by entering Parts of the file names, such as /my* which will list all files and directories starting with “my”. If you only enter a /, then only the directories found will be listed. To apply the Filter, click the Apply Filter button and the Directories and Files lists will be updated and the directory will be listed in the Selection field below as the current selection.
Directories	Selection of directories available to use in the current directory. Single click to place the directory string in the Filter field. Double click to use the directory as the filter (same effect as clicking once and then clicking Apply Filter button), the Directories and Files lists will be updated and the directory will be listed in the Selection field below as the current selection. The sliding controls to the right and bottom of list let you view all available directories.
Files	Single click to select a file.This will insert the file name after the directory listed in the Selection field. This list contains all unfiltered files that are in the filter directory.
Case	
Add...	Specify an additional case. Additional data can be read into another connected Server.
Replace...	Specify a new case to replace an existing case.
Delete	Delete an existing case. Case 1 cannot be deleted, but it can be replaced.
Specify Starting Time Step	Specify starting time step. If not specified, EnSight will load the last step.
Format	Specifies the Format of the dataset. To read EnSight6 data, use the Case format.
(Set) Geometry	Model geometry file name. Clicking button inserts file name shown in Selection field and also inserts path information into Path field. File name can alternatively be typed into field.
(Set) Result	Result file name corresponding to the geometry file. For most data formats this file is optional. Clicking button inserts file name shown in Selection field and also inserts path information into Path field. File name can alternatively be typed into field.
(Set)Measured	Name of a measured file. This is an optional file. Clicking button inserts file name shown in Selection field and also inserts path information into Path field. File name can alternatively be typed into field.
Path	Path to dataset location is inserted by clicking (Set) buttons or may be entered. If blank, files are read from the Server‘s current working directory. Can use the tilde character (~) to specify home directory on the Server host system.
Selection	File or directory selected. Click the appropriate (Set) button to use information in this field.
Okay	Click to read the files specified in the (Set) fields and close the File Selection dialog.
Apply Filter	Click to apply the string in the Filter field.

Cancel — Click to close the File Selection dialog without reading the files specified in the (Set) fields.

(see How To Read EnSight6 Data)

Loading Parts from EnSight6 Data

Data Part Loader dialog for Unstructured EnSight6 Data

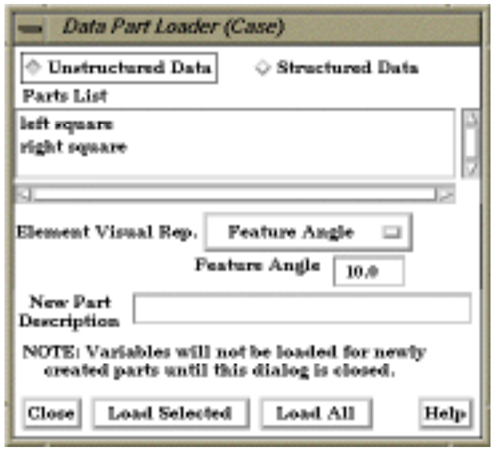

Figure 2-2
Data Part Loader dialog for EnSight6 Unstructured Data

You use the Data Part Loader dialog to control which Parts will be loaded to the Server (and made available on) the EnSight Client. It will automatically open after you have read in data and clicked Okay in the File Selection dialog.
`Access: Main Menu > File > Data (Part Loader)...`

Unstructured Data — This toggle indicates that the Part(s) listed in the Part List is(are) unstructured.

Parts List — Lists all unstructured EnSight6 format Parts in the data files which may be loaded to the Server (and subsequently to the Client). An EnSight6 data file can have unstructured, structured, or both types of Parts.

Element Visual Rep. — Parts are defined on the server as a collection of 1, 2, and 3D elements. EnSight can show you all of the faces and edges of all of these elements, but this is usually a little overwhelming, thus EnSight offers several different *Visual Representations* to simplify the view in the graphics window. Note that the Visual Representation only applies to the EnSight client—it has no affect on the data for the EnSight server.

Element Visual Rep.

Figure 2-3
Element Visual Representation pulldown

3D Border, 2D Full — In this mode, you will see all 1D and 2D elements, but only the outside surfaces of 3D elements.

Border — In Border mode all 1D elements will be shown. Only the unique (non-shared) edges of 2D elements and the unique (non-shared) faces of 3D elements will be shown.

Feature Angle
: When EnSight is asked to display a Part in this mode it first calculates the 3D Border, 2D Full representation to create a list of 1D and 2D elements. Next it looks at the angle between neighboring 2D elements. If the angle is above the Angle value specified the shared edge between the two elements is removed. Only 1D elements remain on the EnSight client after this operation.

Bounding Box
: All Part elements are replaced with a bounding box surrounding the Cartesian extent of the elements of the Part.

Full
: In Full Representation mode all 1D and 2D elements will be shown. In addition, all faces of all 3D elements will be shown.

Non Visual
: This specifies that the loaded Part will not be visible in the Graphics Window because it is only loaded on the Server. Visibility can be turned on again later by changing the representation.

Load Selected
: Loads Parts selected in Parts List to EnSight Server. The Parts are subsequently loaded to the EnSight Client using the specified Visual Representation. If Non Visual is specified, the selected Parts will be loaded to the Server, but not to the Client.

Load All
: Loads all Parts in Parts List to EnSight Server. The Parts are subsequently loaded to the EnSight Client using the specified Visual Representation. If Non Visual is specified, the selected Parts will be loaded to the Server, but not to the Client.

Data Part Loader dialog for Structured EnSight6 data

Figure 2-4

Data Part Loader dialog for EnSight6 Structured Data

You use the Data Part Loader dialog to control which structured Parts will be loaded to the EnSight Server (and subsequently to the EnSight Client). It will automatically open after you have read in data and clicked Okay in the File Selection dialog.

`Access: Main Menu > File > Data (Part Loader)...`

Structured Data — This toggle indicates that the Part data listed is structured.

Parts List — Lists all structured Parts on Server which may be loaded to the EnSight Server (and subsequently to the EnSight Client). When one part is highlighted in this list, the Domain and Node Range fields are updated accordingly.

Element Visual Rep. — Parts are defined on the server as a collection of 1, 2, and 3D elements. EnSight can show you all of the faces and edges of all of these elements, but this is usually a little overwhelming, thus EnSight offers several different *Visual Representations* to simplify the view in the graphics window. Note that the Visual Representation only applies to the EnSight client—it has no affect on the data for the EnSight server.

Element Visual Rep.

Figure 2-5
Element Visual Representation pulldown

3D Border, 2D Full — In this mode, you will see all 1D and 2D elements, but only the outside surfaces of 3D elements.

Border — In Border mode all 1D elements will be shown. Only the unique (non-shared) edges of 2D elements will be shown, and only unique (non-shared) faces of 3D elements will be shown.

Feature Angle — When EnSight is asked to display a Part in this mode it first calculates the 3D Border, 2D Full representation to create a list of 1D and 2D elements. Next it looks at the angle between neighboring 2D elements. If the angle is above the Angle value specified the shared edge between the two elements is removed. Only 1D elements remain on the EnSight client after this operation.

Bounding Box — All Part elements are replaced with a bounding box surrounding the Cartesian extent of the elements of the Part.

Full — In Full Representation mode all 1D and 2D elements will be shown. In addition, all faces of all 3D elements will be shown.

Non Visual — This specifies that the loaded Part will not be visible in the Graphics Window because it is already loaded on the EnSight Server. Visibility can be turned on again later by changing the representation.

Domain — Specifies the general iblanking option to use when creating a structured Part. If the model does not have iblanking, InSide will be specified by default.

Inside	Iblank value = 1 region
Outside	Iblank value = 0 region
All	Ignore iblanking and accept all nodes

Using Node Ranges:

From IJK — Specifies the beginning I,J,K values to use when extracting the structured Part, or a portion of it. Must be >= Min value.

To IJK — Specifies the ending I,J,K values to use when extracting the structured Part, or a portion of it. Must be <= Max value.

Step IJK — Specifies the step increment through I,J,K. A Step value of 1 extracts all original data. A Step value of 2 extracts every other node, etc.

Min IJK — Minimum I,J,K values for Part chosen

Max IJK — Maximum I,J,K values for Part chosen

Part Description — Text field into which you can enter a description for the Part

Create And Load Part — Extracts the data from the data files and creates a Part on the Server (and on the Client unless NonVisual has been specified for Representation) based on all information specified in the dialog.

If more than one Part is highlighted, one Part will be Created for each Part highlighted. Each of the parts will be created with that individual Part's own Min and Max values (you will not be able to specify From and To values). The Step value is the only value you will be able to specify and it will be applied to all Parts.

Iblanked Part Creation section of Structured Part Loader dialog

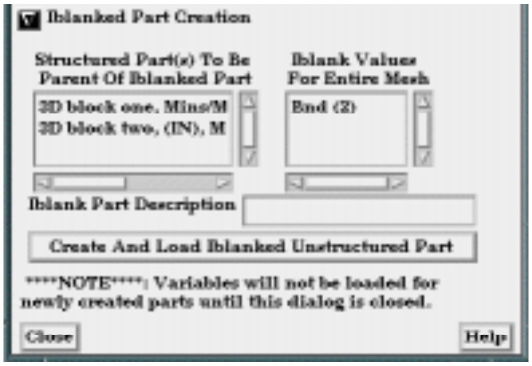

Figure 2-6
Iblanked Part Creation Section of EnSight6 Structured Part Loader dialog

You use this portion of the Part Loader dialog to further extract iblanked regions from structured parts which were created either as inside, outside, or all portions of the model.

Structured Part(s) To Be Parent of Iblanked Part — Lists all structured parts that have been created thus far in the dialog above.

Iblank Values For Entire Mesh — Lists all possible iblank values found in the model. This is a global list and may not apply to all parts.

Iblank Part description — Text field into which you can enter a description for the iblanked part.

Create And Load Iblanked Unstructured Part — Extracts a new iblanked part from an existing structured part. This new part will actually be an unstructured part.

(see How To Read EnSight6 Data)

Reading EnSight5 Data

EnSight5 input data consists of the following files:

- Geometry file (required)
- Result file (optional)
- Variable files (optional)
- Measured/Particle files (optional)

  - Measured/Particle geometry files
  - Measured/Particle results files
  - Measured/Particle variable files

The geometry file is a general finite-element format describing nodes and Parts, each Part being a collection of elements. The result file is a small file allowing the user to name variables and provide time information. The result file points to variable files which contain the scalar or vector information for each node. Measured/Particle files contain data about discrete Particles in space from the simulation code or information directly from actual experimental tests.

EnSight5 data is based on Parts. The Parts defined in the data are always available on the Server. However, all Parts do not have to be loaded to the Client for display. Large flow fields for CFD problems, for example, are needed for computation by the Server, but do not generally need to be seen graphically.

EnSight5 data can have changing geometry, in which case the changing geometry file names are contained in the results file. However, it is still necessary to specify an initial geometry file name.

File Selection dialog

Figure 2-7

File Selection dialog for EnSight5 data

The File Selection dialog is used to specify which files you wish to read.
`Access: Main Menu > File > Data (Reader) …`

Filter — This field specifies the directory name that your data files reside in. Enter a /* at the end of the name to list all of the files and directories contained there. To filter to a smaller file list you can be more specific by entering Parts of the file names, such as /my* which will list all files and directories starting with "my". If you only enter a /, then only the directories found will be listed. To apply the Filter, click the Apply Filter button and the Directories and Files lists will be updated and the directory will be listed in the Selection field below as the current selection.

Directories — Selection of directories available to use in the current directory. Single click to place the directory string in the Filter field. Double click to use the directory as the filter (same effect as clicking once and then clicking Apply Filter button), the Directories and Files lists will be updated and the directory will be listed in the Selection field below as the current selection. The sliding controls to the right and bottom of list let you view all available directories.

Files — Single click to select a file.This will insert the file name after the directory listed in the Selection field. This list contains all unfiltered files that are in the filter directory.

Case

Add... — Specify an additional case. Additional data can be read into another connected Server.

Replace... — Specify a new case to replace an existing case.

Delete — Delete an existing case. Case 1 cannot be deleted, but it can be replaced.

Specify Starting Time Step — Specify starting time step. If not specified, EnSight will load the last step.

Format — Specifies the Format of the dataset.

(Set) Geometry — Model geometry file name. Clicking button inserts file name shown in Selection field and also inserts path information into Path field. File name can alternatively be typed into field.

(Set) Result — Result file name corresponding to the geometry file. For most data formats this file is optional. Clicking button inserts file name shown in Selection field and also inserts path information into Path field. File name can alternatively be typed into field.

(Set)Measured — Name of a measured file. This is an optional file. Clicking button inserts file name shown in Selection field and also inserts path information into Path field. File name can alternatively be typed into field.

Path — Path to dataset location is inserted by clicking (Set) buttons or may be entered. If blank, files are read from the Server's current working directory. Can use the tilde character (~) to specify home directory on the Server host system.

Selection — File or directory selected. Click the appropriate (Set) button to use information in this field.

Okay — Click to read the files specified in the (Set) fields and close the File Selection dialog.

Apply Filter — Click to apply the string in the Filter field.

Cancel Click to close the File Selection dialog without reading the files specified in the (Set) fields.

(see How To Read EnSight5 Data)

Loading Parts from EnSight5 data

Data Part Loader dialog

Figure 2-8
Data Part Loader dialog for EnSight5 data

You use the Data Part Loader dialog to control which Parts will be loaded to the EnSight Server (and subsequently, to the Client). It will automatically open after you have read in data and clicked Okay in the File Selection dialog.
`Access: Main Menu > File > Data (Part Loader)...`

Unstructured Data This toggle indicates that Part data is unstructured. It must be on for EnSight5 format.

Structured Data This toggle is not available for EnSight5 data.

Parts List Lists all Parts in the data files which may be loaded to the EnSight Server (and subsequently, to the Client).

Element Visual Rep. Parts are defined on the server as a collection of 1, 2, and 3D elements. EnSight can show you all of the faces and edges of all of these elements, but this is usually a little overwhelming, thus EnSight offers several different *Visual Representations* to simplify the view in the graphics window. Note that the Visual Representation only applies to the EnSight client—it has no affect on the data for the EnSight server.

Figure 2-9
Element Visual Representation pulldown

3D Border, 2D Full — In this mode, you will see all 1D and 2D elements, but only the outside surfaces of 3D elements.

Border — In Border mode all 1D elements will be shown. Only the unique (non-shared) edges of 2D elements will be shown, and only unique (non-shared) faces of 3D elements will be shown.

Feature Angle — When EnSight is asked to display a Part in this mode it first calculates the 3D Border, 2D Full representation to create with a list of 1D and 2D elements. Next it looks at the angle between neighboring 2D elements. If the angle is above the Angle value specified the shared edge between the two elements is removed. Only 1D elements remain on the EnSight client after this operation.

Bounding Box — All Part elements are replaced with a bounding box surrounding the Cartesian extent of the elements of the Part.

Full — In Full Representation mode all 1D and 2D elements will be shown. In addition, all faces of all 3D elements will be shown.

Non Visual — This specifies that the loaded Part will not be visible in the Graphics Window because it is only loaded on the EnSight Server. Visibility can be turned on again later by changing the representation.

Load Selected — Loads Parts selected in Parts List to EnSight Server. The Parts are subsequently loaded to the EnSight Client using the specified Visual Representation. If Non Visual is specified, the selected Parts will be loaded to the Server, but not to the Client.

Load All — Loads all Parts in Parts List to EnSight Server. The Parts are subsequently loaded to the EnSight Client using the specified Visual Representation. If Non Visual is specified, the selected Parts will be loaded to the Server, but not to the Client.

(see How To Read EnSight5 Data)

Reading ABAQUS Data Files

ABAQUS input data consists of the following files:

- Geometry/Results file (required). This file (the ABAQUS .fil file) contains both the geometry and any requested results. It must be generated in ASCII format. Any desired element based results *must* be nodal averaged.

  OPTIONAL: The .dat file containing element set descriptions (for versions of ABAQUS less than 5.6).

- EnSight Measured/Particle files (optional)

EnSight will read an ASCII .fil file directly. Geometry and commonly used nodal results contained in the file will be read. Additionally, commonly used element based results that are averaged at the nodes will be read. The ABAQUS options: *FILE FORMAT, ASCII, and *EL FILE,POSITION=AVERAGED AT NODES are required.

For version 5.6 (and greater) of ABAQUS, the element sets in the .fil file will be used for creating parts. For versions of ABAQUS less than 5.6, if a .dat file with the same root is present in the same directory as the .fil file, and element set information is present in the .dat file, EnSight will create Parts based on the element sets. Otherwise, all elements are placed in a single Part. The ABAQUS option *PREPRINT,MODEL=YES is used to get elements sets into the .dat file.

The ABAQUS reader currently reads the ASCII format for portability. For large models, this can be somewhat slow.

(see How To Read ABAQUS Data)

Reading ANSYS Result Files

ANSYS input data consists of the following files:

- Geometry and Results file (required). The ANSYS .rst file (or similar results files such as .rfl, .rmq, .rth) contains geometry and results and should be entered in the geometry field of the Data dialog.
- EnSight Measured/Particle files (optional).

EnSight allows you to read the geometry and results data directly from an ANSYS results data file. Not all element types possible in ANSYS can be converted to EnSight format. However, EnSight will handle most practical cases just fine.

Note that certain variables may read slower than others. Displacement, acceleration, and velocity vector variables, as well as nodal solution scalars (pressure, temperature, etc.), read in quickly because they are provided at the nodes directly. Stress and strain variables, on the other hand, can be quite time consuming to read because the process involves:

1. getting the element nodal values,
2. computing principal stresses (or strains),
3. applying equivalent stress (or strain) and/or stress (or strain) intensity equations,
4. if shell elements, using one side or the other (user selectable),
5. averaging the values at shared nodes, and,
6. if higher order elements, averaging to get mid-side node values.

ANSYS data is based on Parts. The Parts defined in the data are always read on the Server. These Parts, however, do not all have to be loaded to the Client for display.

(see How To Read ANSYS Data)

Reading ESTET Data Files

ESTET input data consists of one file that contains all geometry and results information. The ESTET data is a structured grid. The data file is binary.

When reading this data into EnSight, you extract Parts from the mesh interactively based on indices assigned to the nodes in the data. Currently, three domains are possible for extracting Parts: inside, outside, and symmetry plane. As you extract Parts, you can limit the domains according to ranges in I, J and K.

The data can be in rectangular, cylindrical, or curvilinear coordinates. EnSight will interpret and convert properly for any of these types.

Once the desired geometry has been extracted as Parts, you are presented with a list of the results variables contained in the file. There is no way to automatically determine which of the results variables are actually vector components, so you are given the opportunity to build the vectors from the variables. The descriptions usually make this a straightforward process. All variables not used as components to vectors are assumed to be scalar variables.

ESTET Vector Builder and Data Part Loader dialogs

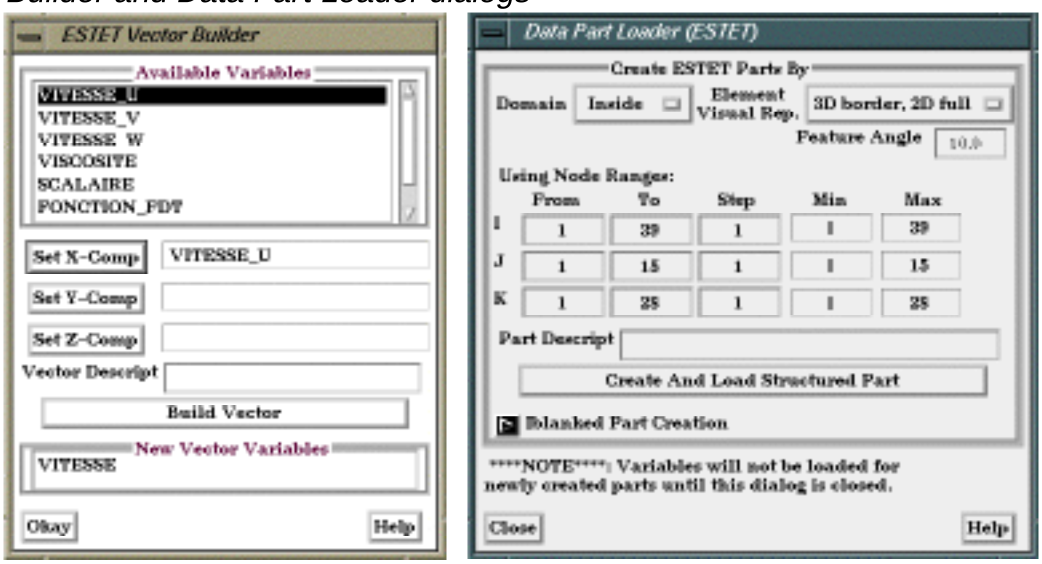

Figure 2-10

ESTET Vector Builder and Data Part Loader dialogs

You use the File Selection dialog to read ESTET data files, the ESTET Vector Builder dialog to build vector variables from scalar components for an ESTET dataset, and the Data Part Loader dialog to extract Parts from an ESTET dataset. The latter two dialogs open in sequence automatically after you click Okay in the File Selection dialog.

`Access: Main Menu > File > Data (Reader)…> ESTET`

Create ESTET Parts By

Domain	Select domain to extract Part.
Inside	Create a structured Part that contains elements whose nodes are flagged as being “inside”.
Outside	Create a structured Part that contains elements whose nodes are flagged as being “outside”.
All	Create a structured Part that contains all elements because node iblanking is ignored.

Using Node Ranges: Specification of node range when creating a Part. Values must be between Min and Max.

Part Descrip Set the name of the Part. If empty, EnSight will assign a name.

Create Part Click to create a Part.

Available Variables Selection to specify a variable to use for the next Set...Comp action.

Set X-Comp Click to set the current selection to be the X component of the vector to build.
Set Y-Comp Click to set the current selection to be the Y component of the vector to build.
Set Z-Comp Click to set the current selection to be the Z component of the vector to build.

Vector Descript Set the name of the vector variable.

Build Vector Click to define the vector variable.

New Vector Variables List of vector variables that have been defined.

Okay Click Okay to load the variable information.

WARNING: You should build all the vectors you are going to use before clicking Okay, because you cannot return to this dialog. If you fail at this point to make all of the vectors desired, it is possible to do so later using the Make Vector function (see Section 4.3, Variable Creation)

Iblanked Part Creation section of Data Part Loader (ESTET) dialog

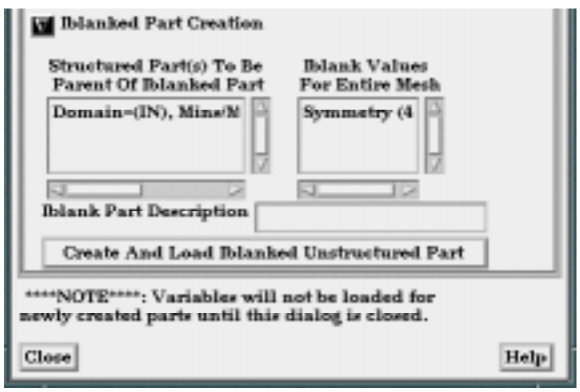

Figure 2-11
Iblanked Part Creation Section of Data Part Loader (ESTET) dialog

You use this portion of the Part Loader dialog to further extract iblanked regions from structured parts which were created either as inside, outside, or all portions of the model.

Structured Part(s) To Be Parent of Iblanked Part Lists all structured parts that have been created thus far in the dialog above.

Iblank Values For Entire Mesh Lists all possible iblank values found in the model. This is a global list and may not apply to all parts.

Iblank Part description Text field into which you can enter a description for the iblanked part.

Create And Load Iblanked Unstructured Part Extracts a new iblanked part from an existing structured part. This new part will actually be an unstructured part.

(see How To Read ESTET Data)

Reading FAST UNSTRUCTURED Data Files

FAST UNSTRUCTURED input data consists of the following files:

- Geometry file (required) This is the FAST UNSTRUCTURED single zone grid file.
- Modified Result file (optional)
- Variable files (optional) These are either a PLOT3D solution file (Q-file) or FAST function file with I = number of points and J=K=1.
- EnSight Measured/Particle files (optional)

FAST UNSTRUCTURED is a format containing triangle and/or tetrahedron elements. The triangles have tags indicating a grouping for specific purposes. EnSight will read the unstructured single zone grid format for this data type, placing all tetrahedral elements into the first Part, and the various triangle element groupings into their own Parts.

The modified EnSight results file allows results to be read from PLOT3D-like Q-files or FAST-like function files. They can be time dependent.

FAST UNSTRUCTURED data can have changing geometry. When this is the case, the changing geometry file names are contained in the results file. However, it is still necessary to specify an initial geometry file name.

(see How To Read FAST Unstructured Data)

Reading FIDAP Neutral Files

A FIDAP Neutral file contains all of the necessary geometry and result information for use with EnSight.

FIDAP data is based on Parts. The Parts defined in the data are always read on the Server. They do not, however, all have to be loaded to the Client for display. Large flow fields for CFD problems, for example, are needed for computation by the Server, but do not generally need to be seen graphically.

EnSight measured/Particle files can also be read with a FIDAP model.

(see How To Read FIDAP Neutral Data)

Reading FLUENT Universal Files

The FLUENT Universal file contains all of the necessary geometry and result information for use with EnSight for a steady-state case. If the case is transient, EnSight needs a Universal file for each time step of the analysis and a modified version of the EnSight results file.

FLUENT data is based on Parts. The Parts defined in the data are always read on the Server. They do not, however, all have to be loaded to the Client for display. Large flow fields for CFD problems, for example, are needed for computation by the Server, but do not generally need to be seen graphically.

EnSight measured/Particle files can also be read with a FLUENT model.

(see How To Read FLUENT Universal Data)

Reading Movie.BYU Files

Movie.BYU input data consists of the following files:

- Geometry file (required)
- Results file (optional)
- Variable files (optional)
- EnSight Measured/Particle files (optional).

Movie.BYU has a general n-sided polygon data format. In translating this format to the element-based EnSight data format, not all elements possible in the Movie.BYU format can be converted to EnSight format. However, for most practical cases there are no problems.

Movie.BYU datasets can be read directly by EnSight. Additionally, an external translator, “movieto5”, is provided if you wish to convert the actual data files to EnSight format.

In order to read Movie.BYU data result files into EnSight, you must create a results file of the same format as EnSight. The external translator, “mpgs4to5,” can be used to generate a results file if you do not want to create your own using a text editor.

Movie.BYU data is based on Parts. The Parts defined in the data are always read on the Server. They do not, however, all have to be loaded to the Client for display.

Movie.BYU data can have changing geometry. When this is the case, the changing geometry file names are contained in the results file. However, it is still necessary to specify an initial geometry file name.

(see How To Read MOVIE.BYU Data)

Reading MPGS4 Data Files

MPGS4 data files consist of the following:

- Geometry file (required)
- EnSight format Results file (optional)
- Variable files (optional)
- EnSight Measured/Particle files (optional).

MPGS4.x uses a general n-sided polygon, n-faced polyhedral data format. In going from this format to the specific element data format of EnSight, you encounter the problem associated with translating from a general format to a specific format. Not all elements possible in MPGS4.x can be converted to EnSight format. However, there will not be a problem in most situations.

MPGS4.x models of modest size can be read directly into EnSight. Size can become an issue since the amount of memory needed to do the conversion in EnSight to the internal data format in a reasonable length of time can become excessive for large models. An external translator, “mpgs4to5”, is provided for the larger models. You should also consider using the external translator to convert MPGS4.x data to EnSight data if you need to continue loading the same dataset, as this will perform the data conversion one time while reading it into EnSight will continue to take resources each time the data is read. Converting it from MPGS4.x to EnSight format also has the advantage of taking less disk space as the EnSight format is more compact.

In order to read MPGS4.x results directly into EnSight, you must create a results file of the same format as EnSight. The external translator, “mpgs4to5,” can be used to generate a results file if you do not want to create your own using an editor.

MPGS4.x data is based on Parts. The Parts defined in the data are always read on the Server. They do not, however, all have to be loaded to the Client for display. Large flow fields for CFD problems, for example, are needed for computation on the Server, but do not generally need to be seen graphically.

MPGS4.x data can have changing geometry. When this is the case, the changing geometry file names are contained in the results file. However, it is still necessary to specify an initial geometry file name.

(see How To Read MPGS Data)

Reading N3S Data Files

N3S input data consists of the following files:

- Geometry file (required)
- Results file (required)
- EnSight Measured/Particle files (optional).

N3S is a data format developed by Electricité de France (EDF) consisting of a geometry file and a results file. For this data format, both files are always required. Versions 3.0 and 3.1 are both supported.

When reading N3S data into EnSight, you extract Parts from the mesh interactively based on different color numbers or boundary conditions. The available color numbers and boundary conditions for the model are presented.

N3S Part Creator dialog

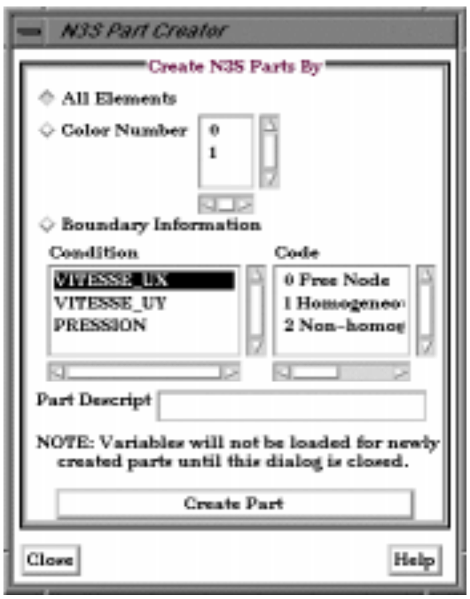

Figure 2-12
N3S Part Creator dialog

You use the File Selection dialog to read in N3S dataset files. You use the N3S Part Creator dialog to extract Parts from a N3S dataset.
`Access: Main Menu > File > Data (Reader) …> N3S`

Create N3S Parts By

All Elements — Selection to create a Part using all of the elements available within the data file.

Color Number — Selection to create a Part according to the color number associated with each element.

Boundary Information — Selection to create a Part according to specified conditions and codes.

Condition — Select boundary condition to use for Part creation.

Code — Select Code to use for boundary condition.

Part Descript — Specify name for Part.

Create Part — Click to create a Part. The Part is listed in the main Parts list of the Parts & Frames dialog and is displayed in the Main View window.

(see How To Read N3S Data)

Reading PLOT3D Data Files

PLOT3D is a commonly used structured data format and input data consists of the following files:

- Geometry file. This is a required file. (Structured GRID file with FAST enhancements)
- Modified EnSight Results file (optional). A standard plot3d Q-file can be read in the results field in place of a modified EnSight Results file.
- Variable files, which are solution (PLOT3D) or function (FAST) files (optional)
- EnSight Measured/Particle files (optional).

When reading PLOT3D files into EnSight, you extract Parts from the mesh based on a domain, a list of zones, and/or indices assigned to the nodes in the data. Currently, three domains are possible for extracting structured Parts: (a) inside, (b) outside, or (c) all. These options are dependent on what the file format is from the parameters defined in the previous paragraph. For instance, when using single zone, non-iblanked data the domain is fixed at “Inside” and the one zone listed in the zones list is selected. As you extract Parts from a single zone file, however, it is possible to limit the domains according to ranges in I, J, and K.

Once the desired structured Parts have been extracted from the geometry, further iblanking options can be used to extract unstructured parts, such as for boundaries. When the Data Reader (PLOT3D) dismissed, the user is presented with a list of the result variables available from the result file

To successfully read PLOT3D data, the following information must be known about the data:

1. format - ASCII, C binary, or Fortran binary
2. whether single or multizone
3. dimension - 3D, 2D, or 1D
4. whether iblanked or not
5. precision - single or double

EnSight attempts to determine these five settings automatically from the grid file. The settings that were determined (for the first four) are shown in the Part Builder dialog, where you can override them manually if needed.

The precision setting is not reflected in the dialog, but is echoed in the Server shell window. The q (or function) file precision will by default be set the same as that of the grid file. In the rare case where the automatic detection is wrong for the grid file or the precision is different for the q (or function) file than for the grid file, commands can be entered into the Command dialog to manually set the precision.

test:	`plot3d_grid_single`	to read grid file as single precision
test:	`plot3d_grid_double`	to read grid file as double precision
test:	`plot3d_qr_single`	to read q (or function) file as single precision
test:	`plot3d_qr_double`	to read q (or function) file as double precision

PLOT3D Part Loader dialog

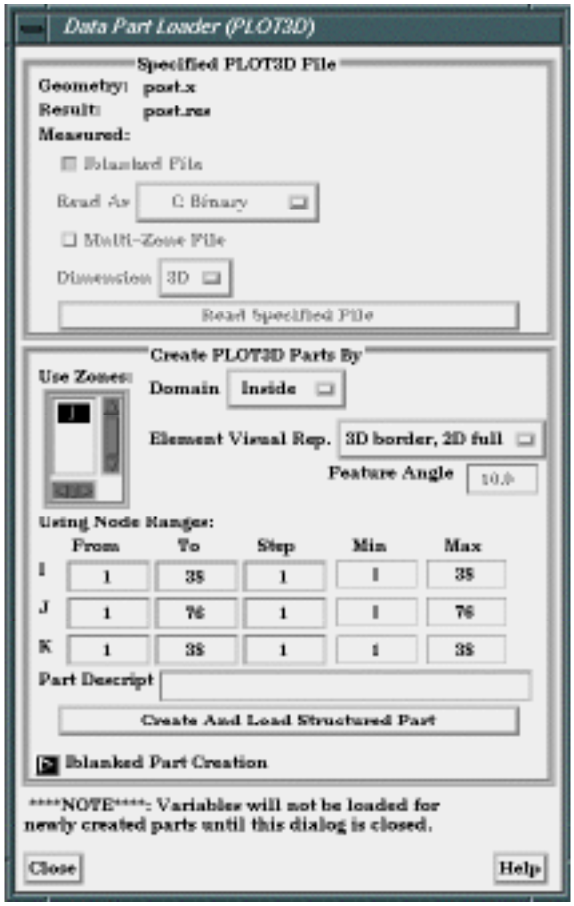

Figure 2-13

Part Data Loader (PLOT3D)

You use the Part Data Loader (PLOT3D) dialog to read a specified PLOT3D file and to extract parts out of the PLOT3D geometry.

`Access: Main Menu > File > Data (Reader)... > PLOT3D`

Geometry Specifies name of file which geometry data will be read.

Result Specifies name of file which result data will be read.

Measured Specifies name of file which measured/discrete data will be read.

IBlanked File Toggle Turn on if geometry field has iblanking.

Read As Specifies file type. Choices are:
ASCII
C Binary (Note: Files may not be portable across hardware platforms).
FORTRAN Binary (Note: Files may not be portable across hardware platforms).

Multi-Zone File Toggle Turn on if dataset contains multiple zones. If Multi-zoned and you are not doing a “between boundary” domain option (see below), a part can span several zones (see Use Zone list below).

Dimension Specifies the dimension of the dataset. Options are 1D, 2D, or 3D. If multi-zone, the dimension of the problem is forced to be 3D.

Read Specified File Click to initiate the reading process.

Use Zones List of Zones defined in the data that can be used to create Parts. If there are multiple zones you can select one or more of them.

Domain Select the domain to create a structured Part from. Options are:

Inside Create structured Part from grid points flagged with Iblanking = 1
Outside Create structured Part from grid points flagged with Iblanking = 0.
All Create part from all grid points (ignores Iblanking).

Using Node Ranges Node range to use when creating a part. Values must be between the Min and Max shown. Option only allowed when a single zone is selected in Use Zones list. (Those values in the Figure above are only shown as examples and pertain to a specific dataset).

Part Descrip Specify name of Part you wish to create.

Create Part Click to create a Part.

Iblanked Part Creation section of Data Part Loader (PLOT3D) dialog

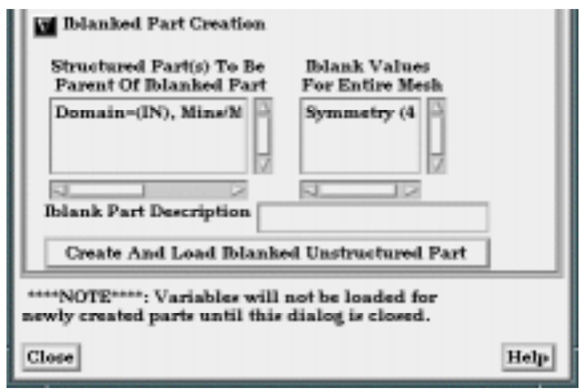

Figure 2-14
Iblanked Part Creation Section of Data Part Loader (PLOT3D) dialog

You use this portion of the Part Loader dialog to further extract iblanked regions from structured parts which were created either as inside, outside, or all portions of the model.

Structured Part(s) To Be Parent of Iblanked Part Lists all structured parts that have been created thus far in the dialog above.

Iblank Values For Entire Mesh Lists all possible iblank values found in the model. This is a global list and may not apply to all parts.

Iblank Part description Text field into which you can enter a description for the iblanked part.

Create And Load Iblanked Unstructured Part Extracts a new iblanked part from an existing structured part. This new part will actually be an unstructured part.

(see How To Read PLOT3D Data)

Reading STAR-CD Files

STAR-CD input data consists of the following files:

- Geometry file, which is the STAR-CD file16 (required)
- Result file, which is either the STAR-CD file9 or file29 (optional)
- EnSight Measured/Particle files (optional)

EnSight allows you to read the geometry and results information from the native STAR-CD files 16, 9, or 29(for STAR-CD version 2.3). EnSight creates Parts based on the table of types for the normal elements and on the types for boundaries. This type information is contained in file 16. These Parts are always loaded on the Server but do not all have to be read to the Client for display. Large flow fields, for example, are needed for computation by the Server but do not generally need to be seen graphically. Some STAR-CD results are based on elements, while others are node based. For those based on elements, an averaging of values to the nodes is performed. A simple contribution averaging scheme is the default, but a distance weighted scheme is also available.

STAR-CD version 3.0 results are currently accessed by converting them to the EnSight format with an external translator provided on the CD.

Note: *STAR-CD version 3.05 provides the capability to directly produce results files in the EnSight format. This is done by using specific EnSight commands in ProStar. In the directory* `$ENSIGHT6_HOME/translators/star-cd` *see the file* `README.DIRECT` *for instructions and example scripts.*

(see How To Read STAR-CD Data)

External Translators

Translators supplied with the EnSight application enable you to use data files from many popular engineering packages. These translators are found in the translators directory on the EnSight distribution tape. A README file is supplied for each translator to help you understand the operation of each Particular translator. These translators are not supported by CEI, but are supplied at no-cost and as source files where possible to allow user modification and porting.

User Defined Readers

A user defined reader capability is included in EnSight which allows otherwise unsupported structured or unstructured data to be read. The user defined reader utilizes a dynamic shared library produced by the user. The definition of the routines for this library is contained in a README file on the CD. Other instructions for producing and using this option are contained therein. In the directory `$ENSIGHT6_HOME//userd_defined` _src/readers
see the file named `README`.

Troubleshooting Loading Data

Problem	Probable Causes	Solutions
Data loads slowly	Loading more Parts than needed	For some models, especially external fluid flow cases, there is a flow field Part which does not need to be visualized. Try eliminating the loading of this Part.
	Too many elements	Make sure the default element representation for Model Parts is set to 3D Border/2D Full before loading the data. In some cases it is helpful to set the representation to Feature Angle before loading.
	Client is swapping because it does not have enough memory to hold all the Parts specified.	Try loading fewer Parts or installing more memory to handle the dataset size.
	Server is swapping because it does not have enough memory to hold all of the Parts contained in the dataset.	Install more memory in your Server host system, reduce the number of variables activated, or somehow reduce the geometry’s size. (If you can get the data in, you can cut away any area not now needed. What is left can then be saved as a geometric entity and that new dataset used for future postprocessing.)
Error reading data	Incorrect path or filename	Reenter the correct information
	Incorrect file permissions	Change the permissions of the relevant directories and files to be readable by you.
	Temporary file space is full	Temporary files are written to the default temporary directory or the directory specified by the environment variable TMPDIR for both the Client and Server. Check file space by using the command “df” and remove unnecessary files from the temporary directory or other full file systems.
	Format of the data is incorrect	Recheck the data against the data format definition.
EnSight format scalar (or vector) data loads, but appears incorrect. Often range of values off by some orders or magnitude.	Scalar (or vector) information not formatted properly in data file	Format the file according to examples listed under EnSight Variable Files in Section 2.5
	Extra white space appended to one or more of the records	Check for and remove any extra white space appended to each record

2.3 Command Files

Command files contain EnSight command language as ASCII text that can be examined and even edited. They can be saved starting at any point and ending at any point during an EnSight session. They can be replayed at any point in an EnSight session. However, *some command sequences require a certain state to exist*, such as connection to the Server, the data read, or a Part created with a Particular Part number.

There are a multitude of applications for command files in EnSight. They include such things as being able to play back an entire EnSight session, easily returning to a standard orientation, connecting to a specific host, creating Particle traces, setting up a keyframe animation, etc. Anything that you will want to be able to repeatedly do is a candidate for a command file. Further, if it is a task that you frequently do, you can turn the command file into a macro (see To Use Macros below).

Default Command File EnSight is always saving a command file referred to as the *default command file*. This default command file receives a default name starting with “ensigAAA” and is written to your /usr/tmp directory (unless you set your TMPDIR environment variable). This command file can be saved (and renamed) when exiting EnSight. The default command file is primarily intended to be a crash recovery aid. If something unforeseen were to prematurely end your EnSight session, you can recover to the last successfully completed command by restarting EnSight and running the default command file.

Documenting Bugs Command files are one of the best ways of documenting any bugs found in the EnSight system. Hopefully that is a rare occasion, but if it occurs, a command file provided to CEI will greatly facilitate the correction of the bug.

Nested Command Files Command files can be nested, which means that if you have a command file that does a specific operation, you can play that command file from any other command file, as long as any prerequisite requirements are completed. This is done by adding the command `play: <filename>` in the command file.

Command dialog

Figure 2-15
Command dialog

You use the Command dialog to control the execution of EnSight command language. The language can be entered by hand, or as is most often the case, played from a file. This dialog also controls the recording of command files as well as Macro path definition.
`Access: Main Menu > File > Command…`

Command History — Displays most recent command language executed (or recorded). Can click on an entry which will bring entry to the Command Entry field.

Command Entry — Command language entry. Enter command and press RETURN. During file playback, next command to be executed is shown here. Any command preceded by a # is a comment line.

Record — Select to start play file recording. Will be prompted for file name. Can simultaneously record and play files. When engaged, all actions in EnSight are recorded to the specified file.

Play — Play a command file by clicking button and entering command language file name. Command play continues as long as there are commands in file, mouse is not in Main View window, an interrupt: command has not been processed, or the Interrupt button has not been pressed.

Interrupt — Interrupt playback of the command file.

Step — Step though commands of play file. File playback must be stopped. Each click will execute next command shown in Command Entry field.

Skip — Skip over the playing command file’s next command (shown in Command Entry box).

Continue — Continue playing interrupted command file.

Delay Between Commands — Set the delay between commands in seconds when playing a command file.

Delay Refresh — When enabled, will cause the EnSight graphics window to refresh only after the playfile processing has completed or has been interrupted by the user.

Reload Macros — Causes the Macro definitions to be reread from the site preferences directory and from the user’s .ensight directory.

Troubleshooting Command Files

This section describes some common errors when running commands. If an error is encountered while playing back a command file you can possibly retype the command or continue without the command.

Problem	Probable Causes	Solutions
Error in command category	Incorrect spelling in the command category	Check and fix spelling
Command does not exist	Incorrect spelling in the command	Check and fix spelling
Error in parameter	Incorrect integer, float, range, or string value parameter	Fix spelling or enter a legal value
Commands do not seem to play	Mouse cursor in the graphics window	Move the mouse cursor out of the Main View window
	Command file was interrupted by an error or an interrupt command	Click continue in the Command dialog

(see How To Record and Play Command Files)

2.4 Saving and Archiving

EnSight enables you to save a number of different files, from window positions to archiving the entire current state of the program. This section lists every file that can be saved with EnSight. Each file has either a description of how to save it, or refers the user to the section which describes this process. Information about where the file is saved (Client host disk or Server host disk) is also provided.

Files That Can Be Saved With EnSight

Description	Type	Format Info	Saved On
Connection Info	ASCII	Section 2.5	Client
Commands	ASCII	Section 3.6	Client
Flipbook pages	binary	not described	Client
Full Backup Archive	binary	not described	both
Geometry & Results	ASCII	Section 2.5	Server
Graphic images	varies	system doc.	Client
Keyframe frames	binary	not described	Client
Function palettes	ASCII	Section 2.5	Client
Plotter Curves	ASCII	plot list	Client
Query Text Info	ASCII	not described	Client
View States	binary	not described	Client
Window States	ASCII	Section 2.5	Client
Preference Information	ASCII	Section 6.6	Client

Saving or Printing an Image

EnSight enables you to save an image of the Main View to a disk file or send it directly to a printer. The choice of save file formats depends on the implementation, but in all cases it is possible to obtain formats compatible with printers and plotters. Currently Apple PICT, PCL, PostScript, SGI RGB, JPEG, and TARGA formats are available.

EnSight also enables you to save images of an animation to disk files. These files can then be converted and printed or recorded to video equipment (see Section 7.15, Keyframe Animation).

Print/Save Image dialog

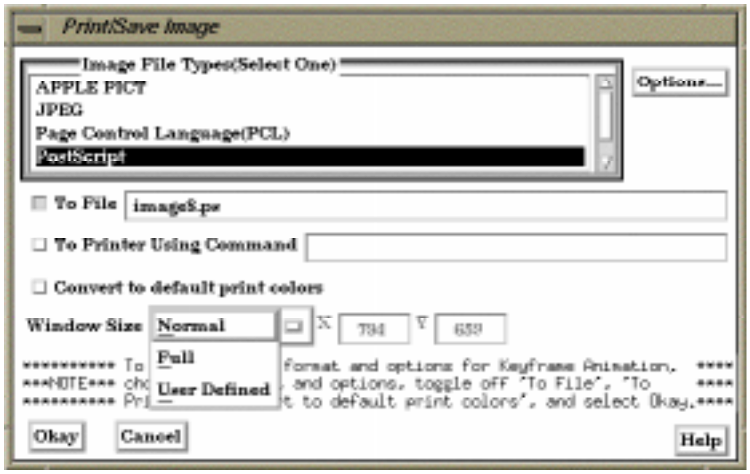

Figure 2-16
Print/Save Image dialog

You use the Print/Save Image dialog to specify the format and destination of an image to save. The destination can be a disk file or a printer. You also access the Image Format Options dialog for the various types from this dialog.
`Access: Main Menu > File > Print/Save Image…`

Image File Types	Click to select image format.
To File Toggle/Field	The image will be saved to this disk file name if toggle is on.
To Printer Using Command Toggle/Field	The command to send a file to the printer if toggle is on
Convert to default print colors	Clicking this toggle on will convert all black to white and all white to black but will leave all other colors as they are.
Window Size	Specifies the size of the Graphics Window and the resulting image size.
Normal	Creates a window which is the size of the current Graphics Window.
Full	Creates a window which is the size of the full screen.
User Defined	Creates a window which is specified in terms of its width and height in the X and Y fields.

Options... Click to set output options for the format selected in Image File Types.

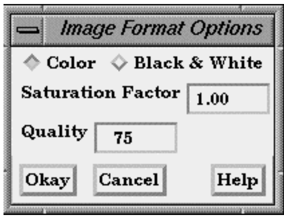

Figure 2-17
Image Format Options dialog for JPEG

Image Format Options dialog for Apple PICT, RGB, and TARGA

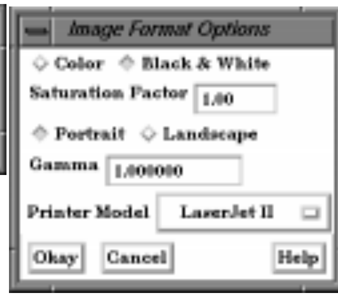

Image Format Options dialog for PCL

Image Format Options dialog for Postscript

Color/Black & White Color versus Black and White toggle

Saturation Factor (all formats) At a value of 1.0, no change to the image. At lower values, a proportionate amount of white is added to each pixel. At a value of 0.0, the image would be all white.

Quality (JPEG) Specifies trade-off between fidelity and compression. 100 maximum fidelity; 0 maximum compression.

Portrait/Landscape (PCL, Postscript) Page Orientation for printing.

Gamma (PCL) Gamma correction factor.

Full Page Scale Factor (Postscript) The percentage of full page scaling to do. This is according to Orientation as well. Values are from 0.0 to 1.0.

Printer Model (PCL) The destination PCL printer model.

Type (Postscript) Type of Postscript output: Move/Draw (vector) or Image Pixels. If type is Image Pixels, shaded 3D objects will be output as pixels while overlay graphics (annotation text, plots, color legends) will be output Move/Draw for higher print quality.

Encapsulated Toggle Generate Encapsulated PostScript (EPS) for importing into other applications. (The graphic typically will appear as a gray box in the importing application on all systems unless the Windows(PC) Preview Capable toggle is also On).

Windows(PC) Preview Capable Toggle Create an Encapsulated PostScript (EPS) file which also has a preview image for use in Windows® applications. (The graphic will still appear as a gray box in the importing application on Macintosh systems).

Element Visibility Culling Toggle (Postscript) Hidden geometry will be removed from the output stream if toggle is on. Valid for Move/Draw output only. On by default.

Element Subdivision (Postscript) Subdivide output primitives (lines and polygons) if toggle is on. Although the output file will be larger, the color distribution will be far superior.Valid for Move/Draw output only. On by default.

Troubleshooting Saving an Image

Problem	Probable Causes	Solutions
Image has blotches or ghosts of other windows in it	A viewport or menu was popped in front of the Main Graphics Window as the image was being saved.	Do not perform any window manager functions until image is finished recording to disk file.
Error while saving image file	Directory or file specified is not writable	Rename the file or change the permissions.
	Ran out of disk space	Check the file system you are writing to with the “df” command then remove any unnecessary files to free up disk space.
	Image format not selected	Select an image format before saving.
Image looks bad when printed	Original on-screen image has low resolution	Make the graphics window as large as possible before saving the image to increase the number of RGB pixels used on the display.
	Image has been dithered during processing	Do not enlarge or reduce the image until it is in your word processor.
	Non-integral ratio of printer resolution to image resolution at final size	The image is a pixel-map image. For best results, the number of printer-dots per image-dot should be an integer. For example, if the original image resolution is 72 dpi, reduced to 48% the final-size resolution is 72/.48 = 150 dpi. On a 600 dpi printer, each image pixel is exactly 4 printer-dots on a side.
Move/Draw PostScript output doesn’t look correct.	Primitives in Move/Draw PostScript output sometimes suffer from sorting problems. (This will be fixed in a subsequent release.)	Use Image Pixel type instead of Move/Draw.

(see How To Print/Save an Image)

Saving and restoring a Full backup

The current state of the EnSight Client and Server host systems may be saved to files. An EnSight session may then be restored to this saved state after restarting at a later time. A Full Backup consists of the following files. First, a small archive information file is created containing the location and name of the Client & Server files that will be described next. Second, a file is created on the Client host system containing the entire state of the Client. Third, a file is created on each Server containing the entire state of that Server. You have control over the name and location of the first file, but only the directories for the other files.

Restoring EnSight to a previously saved state will leave the system in exactly the state EnSight was in at the time of the backup. For a restore to be successful, it is important that EnSight be in a “clean” state. This means that no data can be read in before performing a restore. During a restore, any auto connections to the Server(s) will be made for you. If manual connections were originally used, you will need to once again make them during the restore. (If more than one case was present when the archive was saved, then connection to all the Servers is necessary).

An alternative to a Full Backup is to record a command file up to the state the user wishes to restore at a later date, and then simply replaying the command file. However, this requires execution of the entire command file to get to the restart point. A Full Backup returns you right to the restart point without having to recompute any previous actions.

A Full Backup restores very quickly. If you have very large datasets that take a significant time to read, consider reading them and then immediately writing a Full Backup file. Then, use the Full Backup file for subsequent session instead of reading the data.

Save Full Backup Archive dialog

Figure 2-18
Save Full Backup Archive dialog

You use the Save Full Backup Archive dialog to control the files necessary to perform a full archive on EnSight.
`Access: Main Menu > File > Backup > Save Full Backup…`

Archive Information File Specifies name of Full Backup control file.

Client Directory Specifies the directory for the Client archive file.

Server Directory Specifies the directory for the Server archive file.

Archive information File... Click to display the file selection dialog for specifying the Archive Information File.

Client Directory... Click to display the file selection dialog for specifying the Client Directory.

Server Directory... Click to display the file selection dialog for specifying the Server Directory (for the selected case if there is more than one). Choose a common path if there is more than one.

Okay Click to perform the full backup.

NOTE: This command is written to the command file, but is preceded with a # (the comment character). To make the archive command occur when you play the command file back, uncomment the #.

(see How To Save and Restore an Archive)

File Selection for Restarting from an Archive

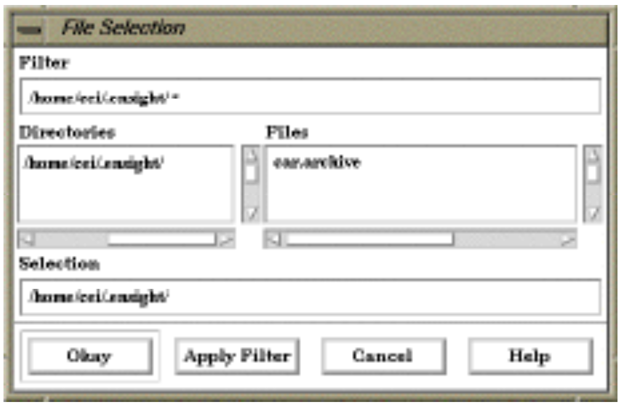

Figure 2-19

File Selection for Restarting from an Archive

You use the Restore Full Archive Backup dialog to read and restore a previously stored archive file.
`Access: Main Menu > File > Backup > Restore Full…`

(see How To Save and Restore an Archive)

Troubleshooting Full Backup

Problem	Probable Causes	Solutions
Error message indicating that all dialogs must be dismissed	When saving and restoring archives, all EnSight dialogs, except for the Client GUI, must be dismissed to free up any temporary tables that are in use. Temporary tables are not written to the archive files.	Dismiss all the Motif dialogs except the main Client GUI.
Backup fails for any reason	Ran out of disk space on the Client or Server host system	Check the file system you are writing to, on both the Server and the Client host systems, with the command “df” then remove any unnecessary files to free up disk space.
	Directory specified is not writable	Rename the directory or change the permissions.

Saving Geometric Entities

Sometimes you may wish to output geometric information from EnSight to be included in a different analysis code, or be part of a HTML based presentation.

EnSight allows you to save the geometric information in Case(EnSight6), VRML, or Case(EnSight Gold) (if you have a Gold license) formats. If you choose to save the geometric information in one of the EnSight Case formats, the files will be written on the Server in either ASCII or binary format. You may also choose to save multiple time steps. The Case file is also created. The geometry, as well as all of the active variables will be saved to files. This feature is limited to saving the following Part types: Model, 2D-Clips, Elevated Surfaces, Developed Surfaces, and Isosurfaces.

The other choice, which is to save the information in a VRML formatted file will allow you to save all of the visible Parts in their current visual state except for Parts which have limit fringes set to transparent. The VRML file will be saved on the Client.

Save Geometric Entities dialog

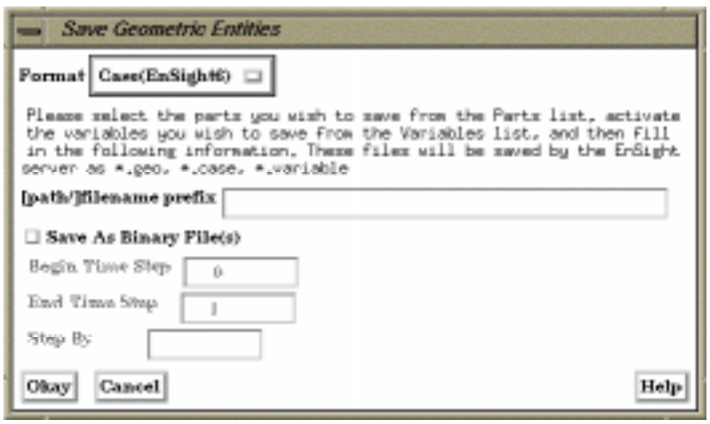

Figure 2-20
Save Geometric Entities dialog

The Save Geometric Entities dialog is used to save Selected Model, 2D-Clip, Isosurface, Elevated Surface, and Developed Surface Parts as EnSight6 files. Thus modified model Parts and certain classes of created Parts can become model Parts of a new dataset. All geometric and variable information is saved for all indicated time steps in either EnSight6 ASCII or binary format.
`Access: Main Menu > File > Save > Save Geometric Entities…`

Format Specify the desired format: Case(EnSight6), Case(EnSight Gold) or VRML.

[path]/filename prefix Specify path and filename prefix name for the saved files. The saved geometry file will be named filename.geo, the result file will be filename.res, and the active variables will be filename.variablename. The VRML file will be filename.wrl.

Save As Binary File(s) Save as Binary File(s) specifies whether to save the data in EnSight6 ASCII (button toggled off - default) or binary (button toggled on) format.

Begin Time Step Begin Time Step field specifies the initial time step for which information will be saved for all selected Parts and activated variables.

End Time Step	End Time Step field specifies the final time step for which information will be saved for all selected Parts and activated variables.
Step By	Step By field specifies the time step increment for which information will be saved for all selected Parts and activated variables starting with Begin Time Step and finishing with End Time Step. The Step By value MUST be an integer.

Troubleshooting Saving Geometric Entities

Problem	Probable Causes	Solutions
A Part was not saved	User attempted to save an unsupported Part type.	Select only Model, Isosurface, 2D-Clip, and Elevated Surface Parts.
Variable(s) not saved	The variable was not activated or the variable was a constant.	Activate all scalar and vector variables you want saved.
Error saving	File prefix indicates a directory that is not writable or disk is out of space.	Re-specify a writable directory and valid prefix name. Remove unneeded files.

(see How To Save Geometric Entities)

Saving the Default Command File for EnSight Session

The command backup file is automatically written during an EnSight session (unless the you have turned off this feature with a Client command line option). When you select Quit, a prompt will ask if you wish to save and possibly rename this file. This file is a normal EnSight command file and can be played in a future session of EnSight to reproduce every step performed in the current session. If you do not save this temporary file in the manner explained below, it will be deleted automatically for you when you Quit EnSight.

Quit Confirmation dialog

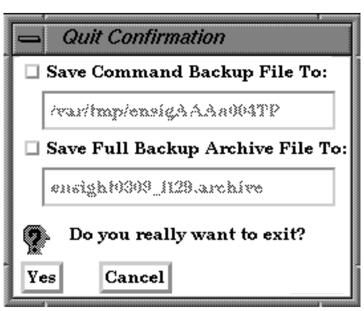

Figure 2-21
Quit Confirmation dialog

You use the Quit Confirmation dialog to save either or both the default command file and an archive file before exiting the program.

`Access: Main Menu > File > Quit…`

Save Command Backup File To: Toggle-on to save the default command file. Can also specify a new name for the command file. Section 2.3, Command Files for more information on using command files.)

Save Full Backup Toggle-on and specify a name to create a Full Backup file.

Yes Click to save the indicated files and terminate the program.

(see How To Record and Play Command Files)

Saving Your EnSight Environment

Every user has different postprocessing needs and personal preferences for how the EnSight windows should be positioned and sized. EnSight allows you to save dialog expandable section settings, and dialog size and position information to a file called “ensight.winpos.default”. EnSight looks for this file at start up (in the current Client directory and if not there in the .ensight directory of the user’s home directory) and will bring the user interface dialogs up according to your saved settings (if the file is found).

Almost all major dialog windows are saved in the “ensight.winpos.default” file. The only exception are minor prompt dialogs. There are also some dialogs for which you cannot save the size (such as the Tool Positions dialog).

The ensight.winpos.default file also contains the size and location for all of the windows containing graphics.

Your mouse button and Icon Bar setting scan also be saved to a user preferences file.

(see Section 6.6, Prefs Menu Functions and How to Save GUI Settings)

2.5 EnSight File Formats

This section describes the format for all readable and writable files in EnSight which you may need access to. The formats described are only for those files that are specific to EnSight. We do not describe data formats not developed by CEI (for example, data formats for various analysis codes). For information about these formats, consult the applicable creator.

Note: *If you are using this documentation to produce your own data translator, please make sure that you follow the instructions exactly as specified. In many cases, EnSight reads data in blocks to improve performance. If the format is not followed, the calculations of how much to read for a block will be thrown off. EnSight does little in the way of error checking data files when they are read. In this respect, EnSight sacrifices robustness for performance.*

As an aid to developing translators, a C library is provided that supports input and output of the native EnSight data format in both ASCII and binary versions. Format Data Files

EnSight6 Format Data Files

EnSight6 data consists of the following files:

- Case (required) (points to all other needed files including model geometry, variables, and possibly measured geometry and variables)

EnSight makes no assumptions regarding the physical significance of the scalar and vector variables. These files can be from any discipline. For example, the scalar file can include such things as pressure, temperature, and stress. The vector file can be velocity, displacement, or any other vector data.

All variable results for EnSight6 are contained in disk files—one variable per file. Additionally, if there are multiple time steps, there must be a set of disk files for each time step.

Sources of EnSight6 data include the following:

- Data that can be translated to conform to the EnSight6 data format
- Data that originates from one of the translators supplied with the EnSight application

The EnSight6 format supports an unstructured defined element set as shown in the figure on the following page. Unstructured data must be defined in this element set. Elements that do not conform to this set must either be subdivided or discarded.

The EnSight6 format also supports a structured block data format which is very similar to the PLOT3D format.

A given EnSight6 model may have either unstructured data, structured data, or a mixture of both.

Supported EnSight Elements

The elements that are supported by the EnSight6 format are:

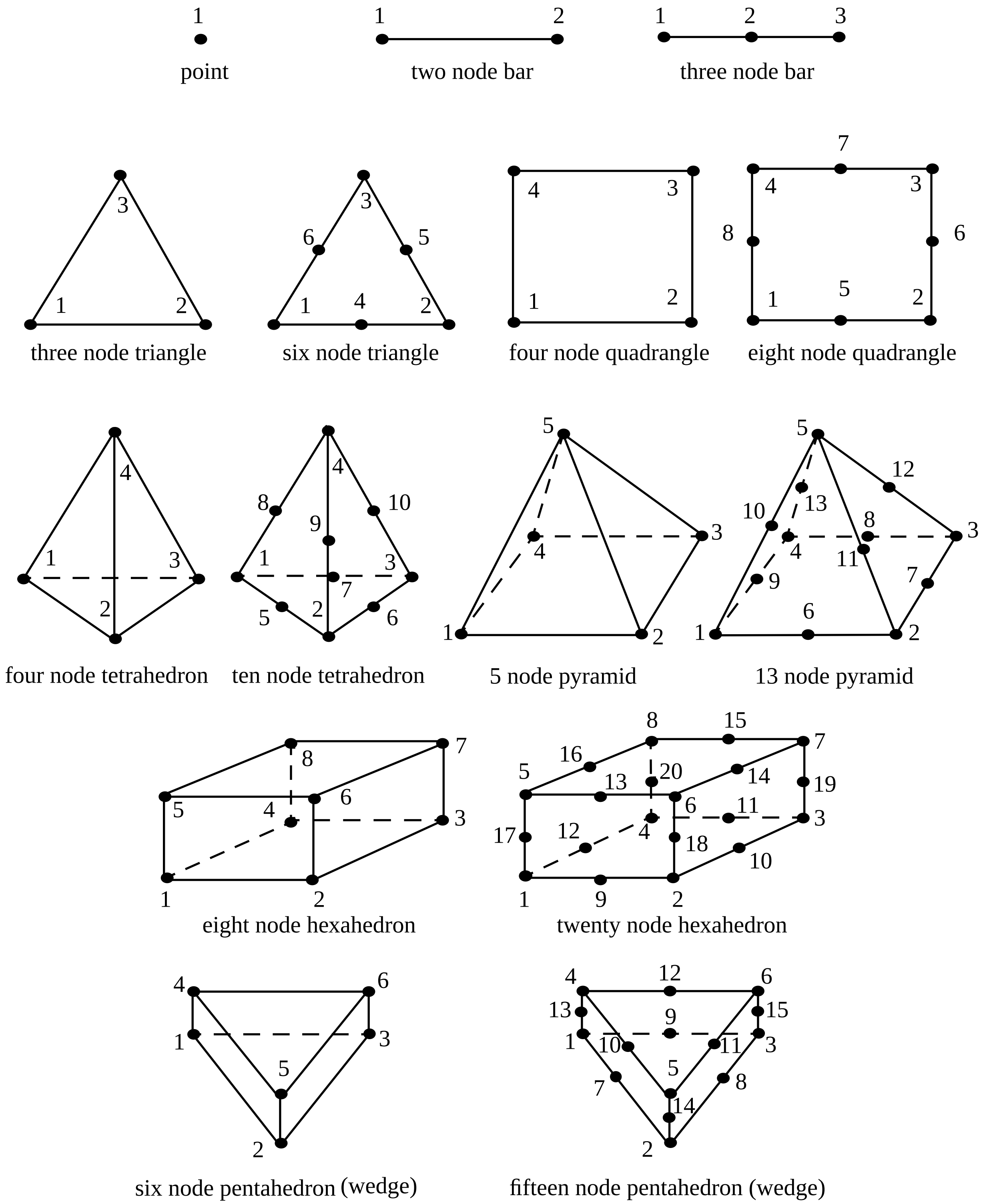

Figure 2-22
Supported EnSight6 Elements

EnSight6 Geometry File Format

The EnSight6 format consists of keywords followed by information. The following seven items are important when working with EnSight6 geometry files:

1. You do not have to assign node IDs. If you do, the element connectivities are based on the node numbers. If you let EnSight assign the node IDs, the nodes are considered to be sequential starting at node 1, and element connectivity is done accordingly. If node IDs are set to off, they are numbered internally; however, you will not be able to display or query on them. If you have node IDs in your data, you can have EnSight ignore them by specifying “node id ignore.” Using this option may reduce some of the memory taken up by the Client and Server, but display and query on the nodes will not be available.
2. You do not need to specify element IDs. If you specify element IDs, or you let EnSight assign them, you can show them on the screen. If they are set to off, you will not be able to show or query on them. If you have element IDs in your data you can have EnSight ignore them by specifying “element id ignore.” Using this option will reduce some of the memory taken up by the Client and Server. This may or may not be a significant amount, and remember that display and query on the elements will not be available.
3. The format of integers and real numbers **must be followed** (See the Geometry Example below).
4. Integers are written out using the following integer format:

From C:	`8d` format
From FORTRAN:	`i8` format

   Real numbers are written out using the following floating-point format:

From C:	`12.5e` format
From FORTRAN:	`e12.5` format

   The number of integers or reals per line must also be followed!
5. By default, a Part is processed to show the outside boundaries. This representation is loaded to the Client host system when the geometry file is read (unless other attributes have been set on the workstation, such as feature angle).
6. Coordinates for unstructured data must be defined before any Parts can be defined. The different elements can be defined in any order (that is, you can define a hexa8 before a bar2).
7. A Part containing structured data cannot contain any unstructured element types or more than one block. **Each structured Part is limited to a single block.** A structured block is indicated by following the Part description line with either the “block” line or the “block iblanked” line. An “iblanked” block must contain an additional integer array of values at each node, traditionally called the iblank array. Valid iblank values for the EnSight format are:

   0 for nodes which are exterior to the model, sometimes called blanked-out nodes
   1 for nodes which are interior to the model, thus in the free stream and to be used
   <0 or >1 for any kind of boundary nodes

In EnSight's structured Part building dialog, the iblank option selected will control which portion of the structured block is "created". Thus, from the same structured block, the interior flow field part as well as a symmetry boundary part could be "created".

Note: By default EnSight does not do any "partial" cell iblank processing. Namely, only complete cells containing no "exterior" nodes are created. It is possible to obtain partial cell processing by issuing the "test:partial_cells_on" command in the Command Dialog before reading the file.

Generic Format

Not all of the lines included in the following generic example file are necessary:

```
description line 1-
description line 2 |
node id <off/given/assign/ignore> |   All geometry files must
element id <off/given/assign/ignore> |   contain these first six lines
coordinates |
# of unstructured nodes-
id x y z
id x y z
id x y z
.
.
.
part #
description line
point
number of points
id nd
id nd
id nd
.
.
.

bar2
number of bar2's
id nd nd
id nd nd
id nd nd
.
.
.
bar3
number of bar3's
id nd nd nd
id nd nd nd
id nd nd nd
.
.
.
tria3
number of three node triangles
id nd nd nd
id nd nd nd
id nd nd nd
.
.
```

```
.
tria6
number of six node triangles
id nd nd nd nd nd nd
.
.
.
quad4
number of quad 4’s
id nd nd nd nd
id nd nd nd nd
id nd nd nd nd
id nd nd nd nd
.
.
.
quad8
number of quad 8’s
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
.
.
.
tetra4
number of 4 node tetrahedrons
id nd nd nd nd
id nd nd nd nd
id nd nd nd nd
id nd nd nd nd
.
.
.
tetra10
number of 10 node tetrahedrons
id nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd
.
.
.
pyramid5
number of 5 node pyramids
id nd nd nd nd nd
id nd nd nd nd nd
id nd nd nd nd nd
id nd nd nd nd nd
.
.
.
pyramid13
number of 13 node pyramids
id nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd
.
.
.
```

```
hexa8
number of 8 node hexahedrons
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
.
.
.
hexa20
number of 20 node hexahedrons
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
.
.
.
penta6
number of 6 node pentahedrons
id nd nd nd nd nd nd
id nd nd nd nd nd nd
id nd nd nd nd nd nd
id nd nd nd nd nd nd
id nd nd nd nd nd nd
.
.
.
penta15
number of 15 node pentahedrons
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
.
.
.

part #
description line
block                                 #mm=i*j*k
i j k
x_m1 x_m2 x_m3 ..... x_mm             (6/line)
y_m1 y_m2 y_m3 ..... y_mm              "
z_m1 z_m2 z_m3 ..... z_mm              "

part #
description line
block     iblanked                    #mm=i*j*k
i j k
x_m1 x_m2 x_m3 ..... x_mm             (6/line)
y_m1 y_m2 y_m3 ..... y_mm              "
z_m1 z_m2 z_m3 ..... z_mm              "
ib_m1 ib_m2 ib_m3 .... ib_mm          (10/line)
```

EnSight6 Geometry File Example

The following is an example of the format of an EnSight6 geometry file:

```
this is the first description line of an example problem
this is the second description line
node id given
element id given
coordinates
      10
       5 1.00000e+00 0.00000e+00 0.00000e+00
     100 0.00000e+00 1.00000e+00 0.00000e+00
     200 0.00000e+00 0.00000e+00 1.00000e+00
      40 1.00000e+00 1.00000e+00 0.00000e+00
      22 1.00000e+00 0.00000e+00 1.00000e+00
    1000 2.00000e+00 0.00000e+00 0.00000e+00
      55 0.00000e+00 2.00000e+00 0.00000e+00
      44 0.00000e+00 0.00000e+00 2.00000e+00
     202 2.00000e+00 2.00000e+00 0.00000e+00
     101 2.00000e+00 0.00000e+00 2.00000e+00
part 1
This is Part 1, a very strange Part
tria3
       2
     101     100     200      40
     201     101       5    1000
tetra4
       1
     102     100     202     101    1000
part 2
This is Part 2, it is also strange
bar2
       1
     103     101    1000
part 3
This is Part 3, a block structured Part with iblanking
block iblanked
2 3 2
 0.00000e+00 1.00000e+00 0.00000e+00 1.00000e+00 0.00000e+00 1.00000e+00
 0.00000e+00 1.00000e+00 0.00000e+00 1.00000e+00 0.00000e+00 1.00000e+00
 0.00000e+00 0.00000e+00 1.00000e+00 1.00000e+00 2.00000e+00 2.00000e+00
 0.00000e+00 0.00000e+00 1.00000e+00 1.00000e+00 2.00000e+00 2.00000e+00
 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00 1.00000e+00
       1       1       2       2       1       1       1       1       2       2
       1       1
```

EnSight6 Case File Format

The Case file is an ASCII free format file that contains all the file and name information for accessing model (and measured) geometry, variable, and time information. It is comprised of four sections (FORMAT, GEOMETRY, VARIABLE, TIME) as described below:

Notes: *All lines in the Case file are limited to 79 characters.*

The titles of each section must be in all capital letters.

Anything preceded by a "#" denotes a comment and is ignored. Comments may append information lines or be placed on their own lines.

Information following ":" may be separated by white spaces or tabs.

Format Section This is a required section which specifies the type of data to be read. Currently, "ensight" is the only acceptable type.

Usage:

```
FORMAT
type:     ensight
```

Geometry Section This is a required section which specifies the geometry information for the model (and measured geometry if present).

Usage:

```
GEOMETRY
model:    [ts]      filename  [change_coords_only]
measured: [ts]      filename
```

where ts = time set number as specified in TIME section. This is optional.
filename = The filename of the geometry file
-> If this is a static geometry case, the filename will <u>not</u> contain "*" wildcards.
-> If this is a changing geometry case, the filename will contain "*" wildcards.
change_coords_only = The option to indicate that the changing geometry (as indicated by wildcards in the filename) is coords only. Otherwise, changing geometry connectivity will be assumed.

Variable Section This is an optional section which specifies the files and names of the variables. Constant variable values can also be set in this section.

Usage:

VARIABLE

constant per case:	description	value
scalar per node:	description	filename
vector per node:	description	filename
scalar per element:	description	filename
vector per element:	description	filename
scalar per measured node:	description	filename
vector per measured node:	description	filename

where description = The variable name (ex. Pressure, Temperature, etc.)
filename = The filename of the variable file
-> If this is a steady state case, the filename will <u>not</u> contain "*" wildcards.
-> If this is a transient case, the filename will contain "*" wildcards.

Note: *As many variable description lines as needed may be used.*

Note: *The variable description is limited to 19 characters in the current release. Variable names must not contain any of the following reserved characters:*

(	[	+	@	!	*	$
)	]	-	space	#	^	/

Time Section

This is an optional section for steady state cases, but is required for transient cases. It contains time set information. Shown below is information for one time set. Another time set may be specified for measured data as shown in Case File Example 3 below.

Usage:

```
TIME
time set:              ts
number of steps:       ns
filename start number: fs
filename increment:    fi
time values:           time_1  time_2  .... time_ns
```

where ts = timeset number. This is the number referenced in the GEOMETRY section.
ns = number of transient steps
fs = the number to replace the "*" wildcards in the filenames, for the first step
fi = the increment to fs for subsequent steps
time = the actual time values for each step, each of which must be separated by a white space and which may continue on the next line if needed

Case File Example 1

The following is a minimal Case file for a steady state model with some results.

```
FORMAT
type:  ensight
GEOMETRY
model:  example1.geo
VARIABLE
scalar per node:   Pressure      example1.pres
scalar per node:   Temperature   example1.temp
vector per node:   Velocity      example1.vel
```

Case File Example 2

The following is a Case file for a transient model. The connectivity of the geometry is also changing.

```
FORMAT
type:  ensight

GEOMETRY
model:             1             example2.geo**

VARIABLE
scalar per node:   Stress        example2.scl**
vector per node:   Displacement  example2.dis**

TIME
time set:               1
number of steps:        3
filename start number:  0
filename increment:     1
time values:  1.0  2.0  3.0
```

The following files would be needed for Example 2:

```
example2.geo00     example2.scl00     example2.dis00
example2.geo01     example2.scl01     example2.dis01
example2.geo02     example2.scl02     example2.dis02
```

Case File Example 3

The following is a Case file for a transient model with measured data.

This example has pressure given per element.

```
FORMAT
type:  ensight

GEOMETRY
model:             1             example3.geo*
measured:          2             example3.mgeo**


VARIABLE
constant per case:               Density       .5
scalar per element          1    Pressure      example3.pre*
vector per node:            1    Velocity      example3.vel*
scalar per measured node:   2    Temperature   example3.mtem**
vector per measured node:   2    Velocity      example3.mvel**

TIME
time set:               1
number of steps:        5
filename start number:  1
filename increment:     2
time values:            .1 .2 .3      # This example shows that time
                        .4 .5         # values can be on multiple lines
time set:               2
number of steps:        6
filename start number:  0
filename increment:     2
time values:
.05 .15 .25 .34 .45 .55
```

The following files would be needed for Example 3:

```
example3.geo1      example3.pre1      example3.vel1
example3.geo3      example3.pre3      example3.vel3
example3.geo5      example3.pre5      example3.vel5
example3.geo7      example3.pre7      example3.vel7
example3.geo9      example3.pre9      example3.vel9

example3.mgeo00    example3.mtem00    example3.mvel00
example3.mgeo02    example3.mtem02    example3.mvel02
example3.mgeo04    example3.mtem04    example3.mvel04
example3.mgeo06    example3.mtem06    example3.mvel06
example3.mgeo08    example3.mtem08    example3.mvel08
example3.mgeo10    example3.mtem10    example3.mvel10
```

EnSight6 Wild Card Name Specification

If multiple time steps are involved, the file names must conform to the EnSight wild-card specification. This specification is as follows:

- File names must include numbers that are in ascending order from beginning to end.
- Numbers in the files names must be zero filled if there is more than one significant digit.
- Numbers can be anywhere in the file name.
- When the file name is specified in the EnSight result file, you must replace the numbers in the file with an asterisk(*). The number of asterisks specified is the number of significant digits. The asterisk must occupy the same place as the numbers in the file names.

EnSight6 Variable File Format

EnSight6 variable files can either be per_node or per_element. They cannot be both. However, an EnSight model can have some variables which are per_node and others which are per_element.

EnSight variable files for per_node variables contain certain values for each unstructured node and for each structured node. First comes a single description line. Second comes the value for each unstructured node. If it is a scalar file, there is one value per node, while for vector files there are three values per node. Third comes a part line followed by a line containing the word “block”. Following this line are the values for each node of the structured block output in the same IJK order as the coordinates.

The values **must be written** in the following floating point format (**6 per line** as shown in the examples below):

From C: `12.5e` format

From FORTRAN: `e12.5` format

The format of a per_node variable file is as follows:

- Line 1

  This line is a description line.

- Line 2 through the end of the file contains the values at each node in the model.

A generic example for per_node variables:

```
A description line
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
```

part #

```
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
```

part #

```
*.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
*.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
*.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
```

Per_node Variable Example 1 This example shows a scalar file for a geometry with seven defined unstructured nodes and a 2x2x2 structured Part (Part number 2).

```
These are the pressure values for a small geometry
 1.00000E+00 2.00000E+00 3.00000E+00 4.00000E+00 5.00000E+00 6.00000E+00
 7.00000E+00
 part 2
 block
 4.00000E+00 6.00000E+00 9.00000E+00 8.00000E+00 3.00000E+00 2.00000E+00
 7.00000E+00 1.00000E+00
```

Per_node Variable Example 2 This example shows the vector file for the same geometry.

```
These are the velocity values for a small geometry
 1.00000E+00 1.00000E+00 1.00000E+00 2.00000E+00 2.00000E+00 2.00000E+00
 3.00000E+00 3.00000E+00 3.00000E+00 4.00000E+00 4.00000E+00 4.00000E+00
 5.00000E+00 5.00000E+00 5.00000E+00 6.00000E+00 6.00000E+00 6.00000E+00
 7.00000E+00 7.00000E+00 7.00000E+00
 part 2
 block
 4.00000E+00 6.00000E+00 9.00000E+00 8.00000E+00 3.00000E+00 2.00000E+00
 7.00000E+00 1.00000E+00
 6.00000E+00 9.00000E+00 3.00000E+00 3.00000E+00 4.00000E+00 6.00000E+00
 9.00000E+00 8.00000E+00
 3.00000E+00 2.00000E+00 7.00000E+00 1.00000E+00 6.00000E+00 9.00000E+00
 3.00000E+00 3.00000E+00
```

EnSight variable files for per_element variables contain values for each element of designated types of designated Parts. First comes a single description line. Second comes a Part line. Third comes an element type line and fourth comes the value for each element of that type and part. If it is a scalar variable, there is one value per element, while for vector variables there are three values per element. (The number of elements of the given type are obtained from the corresponding EnSight6 geometry file.)

The values must be written in the following floating point format (6 per line as shown in the examples below):

From C: `12.5e` format

From FORTRAN: `e12.5` format

The format of a per_element variable file is as follows:

- Line 1

  This line is a description line.

- Line 2

  Part line, with part number corresponding to the geometry file.

- Line 3

  Element type line (example: tria3, hexa8, ...)

- Line 4 Repeats until next element type line, part line, or end of file is reached. Lists values for each element of this part and type.

A generic example for per_element variables:

```
A description line
part  #
element type
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
part  #
element type
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
```

Per_element Variable Example 1

This example shows a per_element scalar file for a single Part geometry with eight triangles and 9 quads and block data for a 2x2x2 structured Part.

```
These are the temperature values at elements for a small geometry.
part 1
tria3
1.00000E+00 1.10000E+00 1.20000E+00 1.30000E+00 1.30000E+00 1.20000E+00
1.10000E+00 1.00000E+00
quad4
4.00000E+00 6.00000E+00 9.00000E+00 8.00000E+00 3.00000E+00 2.00000E+00
7.00000E+00 1.00000E+00  5.00000E+00
part 2
block
5.00000E+00
```

Per_element Variable Example 2

This example shows a vector file for a similar model. The only difference is that the triangles and quads have been placed in different Parts and there is no structured Part data.

```
These are the velocity values at elements of a small model.
part 1
tria3
1.0000E+00 1.00000E+00 1.00000E+00 1.10000E+00 1.10000E+00 1.10000E+00
1.2000E+00 1.20000E+00 1.20000E+00 1.30000E+00 1.30000E+00 1.30000E+00
1.3000E+00 1.30000E+00 1.30000E+00 1.20000E+00 1.20000E+00 1.20000E+00
1.1000E+00 1.10000E+00 1.10000E+00 1.00000E+00 1.00000E+00 1.00000E+00
part 2
quad4
4.00000E+00 4.00000E+00 4.00000E+00 8.00000E+00 8.00000E+00 8.00000E+00
6.00000E+00 6.00000E+00 6.00000E+00 3.00000E+00 3.00000E+00 3.00000E+00
9.00000E+00 9.00000E+00 9.00000E+00 8.00000E+00 8.00000E+00 8.00000E+00
4.00000E+00 4.00000E+00 4.00000E+00 2.00000E+00 2.00000E+00 2.00000E+00
6.00000E+00 6.00000E+00 6.00000E+00
```

EnSight6 Measured/Particle File Format

The format of a Measured/Particle geometry file is as follows:

- Line 1

  This line is a description line.

- Line 2

  Indicates that this file contains particle coordinates. The words “`particle coordinates`” should be entered on this line without the quotes.

- Line 3

  Specifies the number of Particles.

- Line 4 through the end of the file.

  Each line contains the ID and the X, Y, and Z coordinates of each Particle. The format of this line is "integer real real real" written out in the following format:

  From C: `%8d%12.5e%12.5e%12.5e` format

  From FORTRAN: `i8, 3e12.5` format

A generic measured/Particle geometry file is as follows:

```
A description line
particle coordinates
#_of_Particles
id xcoord ycoord zcoord
id xcoord ycoord zcoord
id xcoord ycoord zcoord
 .
 .
 .
```

Measured Geometry Example

The following illustrates a measured/Particle file with seven points:

```
This is a simple measured geometry file
particle coordinates
       7
     101 0.00000E+00 0.00000E+00 0.00000E+00
     102 1.00000E+00 0.00000E+00 0.00000E+00
     103 1.00000E+00 1.00000E+00 0.00000E+00
     104 0.00000E+00 1.00000E+00 0.00000E+00
     205 5.00000E-01 0.00000E+00 2.00000E+00
     206 5.00000E-01 1.00000E+00 2.00000E+00
     307 0.00000E+00 0.00000E+00-1.50000E+00
```

Writing EnSight6 Binary Files

This section describes the EnSight6 binary files. This format is used to increase the speed of reading data into EnSight.

For binary files, there is a header that designates the type of binary file. This header is: "C Binary" or "Fortran Binary." This must be the first thing in the geometry file only. The format for the file is then essentially the same format as the ASCII format, with the following exceptions:

The ASCII format puts the node and element ids on the same "line" as the corresponding coordinates. The BINARY format writes all node id's then all coordinates.

The ASCII format puts all element id's of a type within a Part on the same "line" as the corresponding connectivity. The BINARY format writes all the element ids for that type, then all the corresponding connectivities of the elements.

FORTRAN binary files should be created as sequential access unformatted files.

In all the descriptions of binary files that follow, the number on the left end of the line corresponds to the type of write of that line, according to the following code:

1. This is a write of 80 characters to the file:

C example:
```
char buffer[80];
strcpy(buffer,"C Binary");
fwrite(buffer,sizeof(char),80,file_ptr);
```

FORTRAN:
```
character*80   buffer
buffer = "Fortran Binary"
write(10) buffer
```

2. This is a write of a single integer:

   C example: `fwrite(&num_nodes,sizeof(int),1,file_ptr);`

   FORTRAN: `write(10) num_nodes`

3. This is a write of an integer array:

   C example: `fwrite(node_ids,sizeof(int),num_nodes,file_ptr);`

   FORTRAN: `write(10) (node_ids(i),i=1,num_nodes)`

4. This is a write of a float array:

   C example: `fwrite(coords,sizeof(float),3*num_nodes,file_ptr);`

   FORTRAN: `write(10) ((coords(i,j),i=1,3),j=1,num_nodes)`

(NOTE: Coords is a single precision array, double precision will not work!)

EnSight6 Binary Geometry

An EnSight binary geometry file contains information in the following order:

(1) <C Binary/Fortran Binary>
(1) description line 1
(1) description line 2
(1) node id <given/off/assign/ignore>
(1) element id <given/off/assign/ignore>
(1) coordinates
(2) #_of_points
(3) [point_ids]
(4) coordinate_array
(1) part #
(1) description line
(1) element_type
(2) #_of_element_type
(3) [element_ids] for the element_type
(3) connnectivities for the element_type
(1) element_type
(2) #_of_element_type
(3) [element_ids] for the element_type
(3) connnectivities for the element_type
:
(1) part #
(1) description line
(1) element_type

(2) #_of_element_type
(3) [element_ids] for the element_type
(3) connnectivities for the element_type
(1) element_type
(2) #_of_element_type
(3) [element_ids] for the element_type
(3) connnectivities for the element_type
(1) part #
(1) description line
(1) block [iblanked]
(3) i j k
(4) all i coords, all j coords, all k coords
(3) [iblanking]
:

Per_node Binary Scalar An EnSight6 binary scalar file contains information in the following order:

(1) description line
(4) scalar_array for unstructured nodes
(1) part #
(1) block
(4) scalar_array for part's structured nodes

Per_node Binary Vector An EnSight6 binary vector file contains information in the following order:

(1) description line
(4) vector array for unstructured nodes
(1) part #
(1) block
(4) vector_array for part's structured nodes

Per_element Binary Scalar An EnSight6 binary scalar file contains information in the following order:

(1) description line
(1) part #
(1) element type (tria3, quad4, ...)
(4) scalar_array for elements of part and type

Per_element Binary Vector An EnSight6 binary vector file contains information in the following order:

(1) description line
(1) part #
(1) element type (tria3, quad4, ...)
(4) vector_array for elements of part and type

Binary Measured An EnSight6 binary measured/Particle geometry file contains information in the following order:

(1) <C Binary/Fortran Binary>
(1) description line 1
(1) particle coordinates
(2) #_of_points
(3) point_ids
(4) coordinate_array

EnSight “Gold” Format Data Files

EnSight Gold data consists of the following files:

- Case (required) (points to all other needed files including model geometry, variables, and possibly measured geometry and variables)

EnSight makes no assumptions regarding the physical significance of the scalar and vector variables. These files can be from any discipline. For example, the scalar file can include such things as pressure, temperature, and stress. The vector file can be velocity, displacement, or any other vector data.

All variable results for EnSight Gold format are contained in disk files—one variable per file. Additionally, if there are multiple time steps, there must be a set of disk files for each time step.

Sources of EnSight Gold format data include the following:

- Data that can be translated to conform to the EnSight Gold data format (including being written from EnSight itself using the Save Geometric Entities option under File->Save)
- Data that originates from one of the translators supplied with the EnSight application

The EnSight Gold format supports an unstructured defined element set as shown in the figure on the following page. Unstructured data must be defined in this element set. Elements that do not conform to this set must either be subdivided or discarded.

The EnSight Gold format also supports the same structured block data format as EnSight6, which is very similar to the PLOT3D format.

A given EnSight Gold model may have either unstructured data, structured data, or a mixture of both.

This format is somewhat similar to the EnSight6 format, but differs enough to allow for more efficient reading of the data. It is intended for 3D, binary, big data models. It is initially limited to EnSight Gold licenses.

Supported EnSight Gold Elements

The elements that are supported by the EnSight Gold format are:

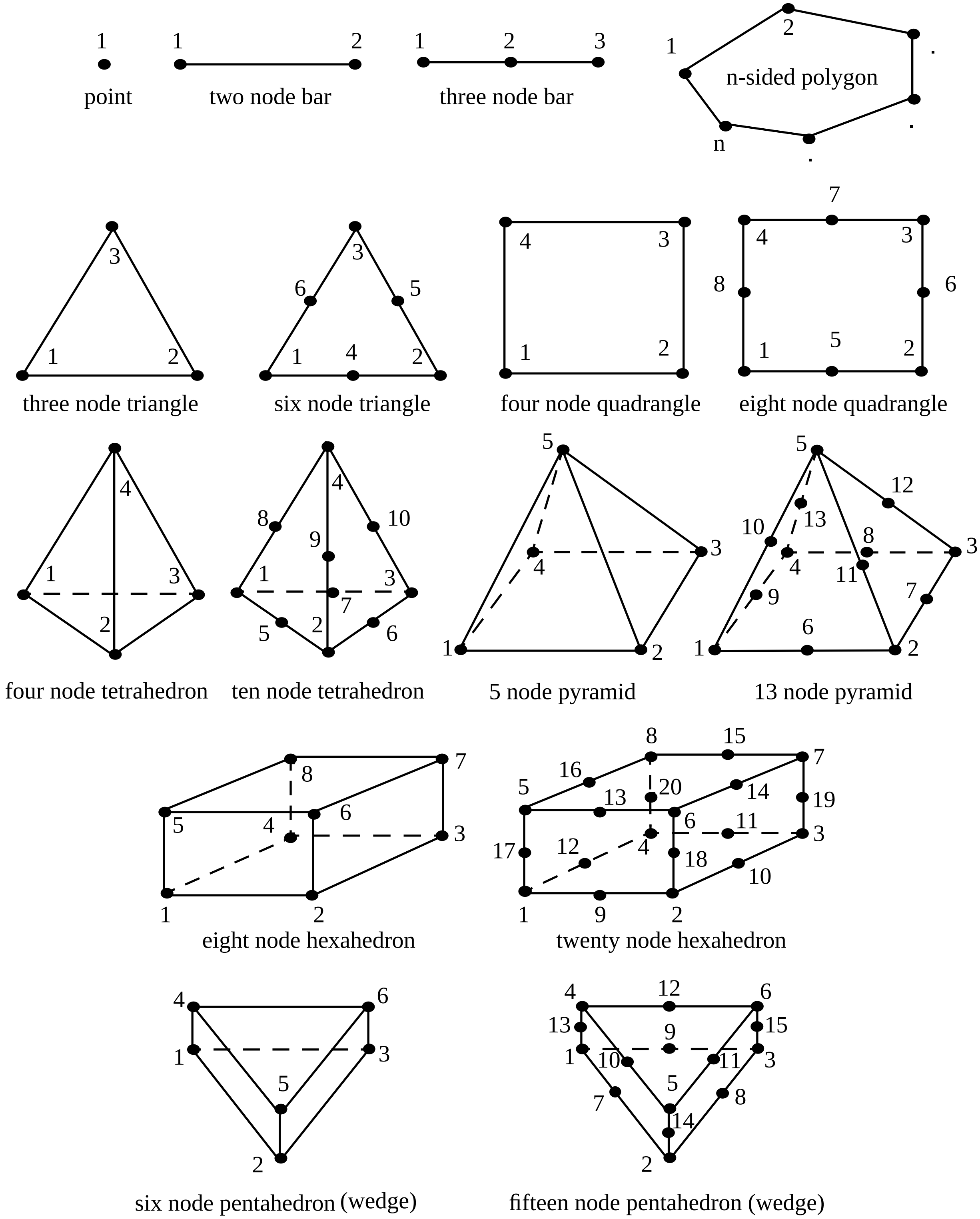

Figure 2-23
Supported EnSight Gold Elements

EnSight Gold Geometry File Format

The EnSight Gold format is part based for both unstructured and structured data. There is no global coordinate array that each part references, but instead - each part contains its own local coordinate array. Thus, the node numbers in element connectivities refer to the coordinate array index, not a node id or label. *This is different than the EnSight6 format!*

The EnSight Gold format consists of keywords followed by information. The following items are important when working with EnSight Gold geometry files:

1. Node ids are optional. In this format they are strictly labels and are not used in the connectivity definition. The element connectivities are based on the local implied node number of the coordinate array in each part, which is sequential starting at one. If you let EnSight assign node IDs, this implied internal numbering is used. If node IDs are set to off, they are numbered internally, however, you will not be able to display or query on them. If you have node IDs given in your data, you can have EnSight ignore them by specifying “node id ignore.” Using this option may reduce some of the memory taken up by the Client and Server, but display and query on the nodes will not be available.
2. Element ids are optional. If you specify element IDs, or you let EnSight assign them, you can show them on the screen. If they are set to off, you will not be able to show or query on them. If you have element IDs given in your data you can have EnSight ignore them by specifying “element id ignore.” Using this option will reduce some of the memory taken up by the Client and Server. This may or may not be a significant amount, and remember that display and query on the elements will not be available.
3. Model extents can be defined in the file so EnSight will not have to determine these while reading in data. If they are not included, EnSight will compute them, but will not actually do so until a dataset query is performed the first time.
4. The format of integers and real numbers **must be followed** (See the Geometry Example below).
5. Integers are written out using the following integer format:

From C:	`10d` format
From FORTRAN:	`i10` format

   Real numbers are written out using the following floating-point format:

From C:	`12.5e` format
From FORTRAN:	`e12.5` format

   The number of integers or reals per line must also be followed!
6. By default, a Part is processed to show the outside boundaries. This representation is loaded to the Client host system when the geometry file is read (unless other attributes have been set on the workstation, such as feature angle).
7. Coordinates for unstructured data must be defined within each part. This is normally done before any element s are defined within a part, but does not

have to be. The different elements can be defined in any order (that is, you can define a hexa8 before a bar2).

8. Parts which contain n-sided polygon elements may not contain other element types. They can however contain several different n-sided sections.

9. A Part containing structured data cannot contain any unstructured element types or more than one block. **Each structured Part is limited to a single block.** A structured block is indicated by following the Part description line with a 'block' line. The various options include:

block	(default is curvilinear)
block rectilinear	(uses i,j,k delta vectors)
block uniform	(uses i,j,k delta values)
block iblanked	
block rectilinear iblanked	
block uniform iblanked	

An "iblanked" block must contain an additional integer array of values at each node, traditionally called the iblank array. Valid iblank values for the EnSight Gold format are:

0 for nodes which are exterior to the model, sometimes called blanked-out nodes
1 for nodes which are interior to the model, thus in the free stream and to be used
<0 or >1 for any kind of boundary nodes

In EnSight's structured Part building dialog, the iblank option selected will control which portion of the structured block is "created". Thus, from the same structured block, the interior flow field part as well as a symmetry boundary part could be "created".

Note: By default EnSight does not do any "partial" cell iblank processing. Namely, only complete cells containing no "exterior" nodes are created. It is possible to obtain partial cell processing by issuing the "test:partial_cells_on" command in the Command Dialog before reading the file.

Generic Format

Usage Notes:

In general a `part` can contain several different `element types`. There are two exceptions to this rule. Parts containing `nsided` polygons cannot contain any other element type, and `block` parts can contain one and only one block.

`element type` can be any of:

`point`	`bar2`	`bar3`
`tria3`	`quad4`	`quad8`
`nsided`	`tetra4`	`tetra10`
`pyramid5`	`pyramid13`	`penta6`
`penta15`	`hexa8`	`hexa20`

`#` = a part number
`nn` = total number of nodes in a part
`ne` = number of elements of a given type
`np` = number of nodes per element for a given element type
`id_*` = node or element id number
`x_*` = x component
`y_*` = y component
`z_*` = z component
`e*_*` = node number for an element
`ib_*` = iblanking value

`[ ]` contain optional portions
`< >` contain choices
`‘` indicates the beginning of an unformatted sequential FORTRAN binary write
`’` indicates the end of an unformatted sequential FORTRAN binary write

C Binary form:

```
C binary
description line 1                                    80 chars
description line 2                                    80 chars
node id <off/given/assign/ignore>                     80 chars
element id <off/given/assign/ignore>                  80 chars
[extents                                              80 chars
xmin xmax ymin ymax zmin zmax]                         6 floats
part                                                  80 chars
#                                                      1 int
description line                                      80 chars
coordinates                                           80 chars
nn                                                     1 int
[id_n1 id_n2 ... id_nn]                               nn ints
x_n1 x_n2 ... x_nn                                    nn floats
y_n1 y_n2 ... y_nn                                    nn floats
z_n1 z_n2 ... z_nn                                    nn floats
element type                                          80 chars
ne                                                     1 int
[id_n1 id_n2 ... id_ne]                               ne ints
e1_n1 e1_n2 ... e1_np
e2_n1 e2_n2 ... e2_np
 .
 .
ne_n1 ne_n2 ... ne_np                                 ne*np ints
element type                                          80 chars
 .
 .
part                                                  80 chars
 .
 .
part                                                  80 chars
#                                                      1 int
description line                                      80 chars
block [iblanked]                                      80 chars
i j k                      # mm = i*j*k                3 ints
x_m1 x_m2 ... x_mm                                    mm floats
y_m1 y_m2 ... y_mm                                    mm floats
z_m1 z_m2 ... z_mm                                    mm floats
[ib_m1 ib_m2 ... ib_mm]                               mm ints
part                                                  80 chars
#                                                      1 int
description line                                      80 chars
block rectilinear [iblanked]                          80 chars
i j k                      # mm = i*j*k                3 ints
x_1 x_2 ... x_i                                       i floats
y_1 y_2 ... y_j                                       j floats
z_1 z_2 ... z_k                                       k floats
[ib_m1 ib_m2 ... ib_mm]                               mm ints
part                                                  80 chars
#                                                      1 int
description line                                      80 chars
block uniform [iblanked]                              80 chars
i j k                      # mm = i*j*k                3 ints
x_origin y_origin z_origin                             3 floats
x_delta y_delta z_delta                                3 floats
[ib_m1 ib_m2 ... ib_mm]                               mm ints
```

Fortran Binary form:

```
'Fortran binary'
'description line 1'                                     80 chars
'description line 2'                                     80 chars
'node id <off/given/assign/ignore>'                      80 chars
'element id <off/given/assign/ignore>'                   80 chars
['extents'                                               80 chars
'xmin xmax ymin ymax zmin zmax']                          6 floats
'part'                                                   80 chars
'#'                                                       1 int
'description line'                                       80 chars
'coordinates'                                            80 chars
'nn'                                                      1 int
['id_n1 id_n2 ... id_nn']                                nn ints
'x_n1 x_n2 ... x_nn'                                     nn floats
'y_n1 y_n2 ... y_nn'                                     nn floats
'z_n1 z_n2 ... z_nn'                                     nn floats
'element type'                                           80 chars
'ne'                                                      1 int
['id_n1 id_n2 ... id_ne']                                ne ints
'e1_n1 e1_n2 ... e1_np
 e2_n1 e2_n2 ... e2_np
 .
 .
 ne_n1 ne_n2 ... ne_np'                                  ne*np ints
'element type'                                           80 chars
 .
 .
'part'                                                   80 chars
 .
 .
'part'                                                   80 chars
'#'                                                       1 int
'description line'                                       80 chars
'block [iblanked]'                                       80 chars
'i j k'                    # mm = i*j*k                   3 ints
'x_m1 x_m2 ... x_mm'                                     mm floats
'y_m1 y_m2 ... y_mm'                                     mm floats
'z_m1 z_m2 ... z_mm'                                     mm floats
['ib_m1 ib_m2 ... ib_mm']                                mm ints
'part'                                                   80 chars
'#'                                                       1 int
'description line'                                       80 chars
'block rectilinear [iblanked]'                           80 chars
'i j k'                    # mm = i*j*k                   3 ints
'x_1 x_2 ... x_i'                                        i floats
'y_1 y_2 ... y_j'                                        j floats
'z_1 z_2 ... z_k'                                        k floats
['ib_m1 ib_m2 ... ib_mm']                                mm ints
'part'                                                   80 chars
'#'                                                       1 int
'description line'                                       80 chars
'block uniform [iblanked]'                               80 chars
'i j k'                    # mm = i*j*k                   3 ints
'x_origin y_origin z_origin                               3 floats
 x_delta y_delta z_delta'                                 3 floats
['ib_m1 ib_m2 ... ib_mm']                                mm ints
```

ASCII form:

```
description line 1                                   A (max of 79 typ)
description line 2                                   A
node id <off/given/assign/ignore>                    A
element id <off/given/assign/ignore>                 A
[extents                                             A
xmin xmax                                            2E12.5
ymin ymax                                            2E12.5
zmin zmax]                                           2E12.5
part                                                 A
#                                                    I10
description line                                     A
coordinates                                          A
nn                                                   I10
[id_n1                                               I10    1/line (nn)
 id_n2
  .
  .
 id_nn]
x_n1                                                 E12.5  1/line (nn)
x_n2
 .
 .
x_nn
y_n1                                                 E12.5  1/line (nn)
y_n2
 .
 .
y_nn
z_n1                                                 E12.5  1/line (nn)
z_n2
 .
 .
z_nn
element type                                         A
ne                                                   I10
[id_n1                                               I10    1/line (ne)
 id_n2
  .
  .
 id_ne]
e1_n1 e1_n2 ... e1_np                                I10    np/line
e2_n1 e2_n2 ... e2_np                                    (ne lines)
 .
 .
ne_n1 ne_n2 ... ne_np
element type                                         A
 .
 .
part                                                 A
 .
 .
part                                                 A
#                                                    I10
description line                                     A
block [iblanked]                                     A
i j k                          # mm = i*j*k          3I10
x_m1                                                 E12.5  1/line (mm)
x_m2
 .
```

```
 .
x_mm
y_m1                                                 E12.5  1/line (mm)
y_m2
 .
 .
y_mm
z_m1                                                 E12.5  1/line (mm)
z_m2
 .
 .
z_mm
[ib_m1                                               I10    1/line (mm)
 ib_m2
  .
  .
 ib_mm]
part                                                 A
#                                                    I10
description line                                     A
block rectilinear [iblanked]                         A
i j k                      # mm = i*j*k              3I10
x_1                                                  E12.5  1/line (i)
x_2
 .
 .
x_i
y_1                                                  E12.5  1/line (j)
y_2
 .
 .
y_j
z_1                                                  E12.5  1/line (k)
z_2
 .
 .
z_k
[ib_m1                                               I10    1/line (mm)
 ib_m2
  .
  .
 ib_mm]
part                                                 A
#                                                    I10
description line                                     A
block uniform [iblanked]                             A
i j k                      # mm = i*j*k              3I10
x_origin                                             E12/5
y_origin                                             E12/5
z_origin                                             E12/5
x_delta                                              E12.5
y_delta                                              E12.5
z_delta                                              E12.5
[ib_m1                                               I10    1/line (mm)
 ib_m2
  .
  .
 ib_mm]
```

Notes:

- If `node id` is `given` or `ignore`, the [id] section must be there for each part.
- If `element id` is `given` or `ignore`, the [id] section must be there for each `element type` of each part
- If `iblanked` is there, the [ib] section must be there for the block.
- x, y, and z coordinates are mandatory, even if a 2D problem.
- If `block rectilinear`, then the x, y, z coordinates change to the x, y, and z delta vectors.
- If `block uniform`, then the x, y, z coordinates change to the x, y, z coordinates of the origin and the x, y, and z delta values.
- Ids are just labels, the coordinate (or element) order is implied.

EnSight Gold Geometry File Example

The following is an example of an ASCII EnSight Gold geometry file:

```
this is the first description line of an example problem
this is the second description line
node id given
element id given
extents
 0.00000e+00 2.00000e+00
 0.00000e+00 2.00000e+00
 0.00000e+00 2.00000e+00
part
         1
This is Part 1, a very strange Part
coordinates
         7           # nn                Do NOT put these # comments in your file!!
         5           # node ids
       100
       200
        40
      1000
       202
       101
 1.00000e+00         # x components
 0.00000e+00
 0.00000e+00
 1.00000e+00
 2.00000e+00
 2.00000e+00
 2.00000e+00
 0.00000e+00         # y components
 1.00000e+00
 0.00000e+00
 1.00000e+00
 0.00000e+00
 2.00000e+00
 0.00000e+00
 0.00000e+00         # z components
 0.00000e+00
 1.00000e+00
 0.00000e+00
```

```
 0.00000e+00
 0.00000e+00
 2.00000e+00
tria3                # element type
         2           # ne
       101           # element ids
       201
         2         3         4
         7         1         5
tetra4
         1
       102
         2         6         7         5
part
         2
This is Part 2, it is also strange
coordinates
         2
       101
      1000
 2.00000e+00
 2.00000e+00
 0.00000e+00
 0.00000e+00
 2.00000e+00
 0.00000e+00
bar2
         1
       103
         1         2
part
         3
This is Part 3, a block structured rectilinear Part with iblanking
block rectilinear iblanked
         2         3         2
 0.00000e+00         # i vector
 1.00000e+00
 0.00000e+00         # j vector
 1.00000e+00
 2.00000e+00
 0.00000e+00         # k vector
 1.00000e+00
         1           # iblanking
         1
         2
         2
         1
         1
         1
         1
         2
         2
         1
         1
```

EnSight Gold Case File Format

The Case file is identical to that of the EnSight6 data format, with one exception - the addition of “gold” to the FORMAT type as shown below.

Format Section This is a required section which specifies the type of data to be read.

Usage:

```
FORMAT
type:     ensight gold
```

EnSight Gold Wild Card Name Specification

If multiple time steps are involved, the file names must conform to the EnSight wild-card specification. This specification is as follows:

- File names must include numbers that are in ascending order from beginning to end.
- Numbers in the files names must be zero filled if there is more than one significant digit.
- Numbers can be anywhere in the file name.
- When the file name is specified in the EnSight case file, you must replace the numbers in the file with an asterisk(*). The number of asterisks specified is the number of significant digits. The asterisk must occupy the same place as the numbers in the file names.

EnSight Gold Variable File Format

EnSight Gold variable files can either be per_node or per_element. They cannot be both. However, an EnSight model can have some variables which are per_node and others which are per_element.

EnSight Gold variable files for per_node variables contain values for each unstructured node and for each structured node. First comes a single description line. Second comes a part line. Third comes a line containing the part number. Fourth comes a ‘coordinates’ line or a ‘block’ line. If a ‘coordinates’ line, the value for each unstructured node of the part follows. If it is a scalar file, there is one value per node, while for vector files there are three values per node (output in the same component order as the coordinates, namely, all x components, then all y components, then all z components). If it is a ‘block’ line, the value(s) for each structured node follows. The values for each node of the structured block are output in the same IJK order as the coordinates. (The number of nodes in the part are obtained from the corresponding EnSight Gold geometry file.)

C Binary form:

SCALAR FILE:

```
description line 1                                    80 chars
part                                                  80 chars
#                                                      1 int
coordinates                                           80 chars
s_n1 s_n2 ... s_nn                                    nn floats
```

```
part                                                80 chars
 .
 .
part                                                80 chars
#                                                    1 int
block                    # mm = i*j*k               80 chars
s_m1 s_m2 ... s_mm                                  mm floats
```

VECTOR FILE:

```
description line 1                                  80 chars
part                                                80 chars
#                                                    1 int
coordinates                                         80 chars
vx_n1 vx_n2 ... vx_nn                               nn floats
vy_n1 vy_n2 ... vy_nn                               nn floats
vz_n1 vz_n2 ... vz_nn                               nn floats
part                                                80 chars
 .
 .
part                                                80 chars
#                                                    1 int
block                    # mm = i*j*k               80 chars
vx_m1 vx_m2 ... vx_mm                               mm floats
vy_m1 vy_m2 ... vy_mm                               mm floats
vz_m1 vz_m2 ... vz_mm                               mm floats
```

Fortran Binary form:

SCALAR FILE:

```
‘description line 1’                                80 chars
‘part’                                              80 chars
‘#’                                                  1 int
‘coordinates’                                       80 chars
‘s_n1 s_n2 ... s_nn’                                nn floats
‘part’                                              80 chars
 .
 .
‘part’                                              80 chars
‘#’                                                  1 int
‘block’                    # mm = i*j*k             80 chars
‘s_m1 s_m2 ... s_mm’                                mm floats
```

VECTOR FILE:

```
‘description line 1’                                80 chars
‘part’                                              80 chars
‘#’                                                  1 int
‘coordinates’                                       80 chars
‘vx_n1 vx_n2 ... vx_nn’                             nn floats
‘vy_n1 vy_n2 ... vy_nn’                             nn floats
‘vz_n1 vz_n2 ... vz_nn’                             nn floats
‘part’                                              80 chars
 .
 .
```

```
'part'                                              80 chars
'#'                                                  1 int
'block'                    # mm = i*j*k             80 chars
'vx_m1 vx_m2 ... vx_mm'                             mm floats
'vy_m1 vy_m2 ... vy_mm'                             mm floats
'vz_m1 vz_m2 ... vz_mm'                             mm floats
```

ASCII form:

SCALAR FILE:

```
description line 1                                  A (max of 79 typ)
part                                                A
#                                                   I10
coordinates                                         A
s_n1                                                E12.5  1/line (nn)
s_n2
 .
 .
s_nn
part                                                A
 .
 .
part                                                A
#                                                   I10
block                       # mm = i*j*k            A
s_m1                                                E12.5  1/line (mm)
s_m2
 .
 .
s_mm
```

VECTOR FILE:

```
description line 1                                  A (max of 79 typ)
part                                                A
#                                                   I10
coordinates                                         A
vx_n1                                               E12.5  1/line (nn)
vx_n2
 .
 .
vx_nn
vy_n1                                               E12.5  1/line (nn)
vy_n2
 .
 .
vy_nn
vz_n1                                               E12.5  1/line (nn)
vz_n2
 .
 .
vz_nn
part                                                A
 .
 .
part                                                A
```

```
#                                             I10
block                   # mm = i*j*k          A
vx_m1                                         E12.5  1/line (mm)
vx_m2
 .
 .
vx_mm
vy_m1                                         E12.5  1/line (mm)
vy_m2
 .
 .
vy_mm
vz_m1                                         E12.5  1/line (mm)
vz_m2
 .
 .
vz_mm
```

Per_node Variable:

Example 1

This example shows a scalar file for a geometry with seven defined unstructured nodes in a part and a 2x2x2 structured Part (Part number 2).

```
These are the pressure values for a small geometry
part
         1
coordinates
block
 1.00000E+00
 2.00000E+00
 3.00000E+00
 4.00000E+00
 5.00000E+00
 6.00000E+00
 7.00000E+00
part
         2
block
 4.00000E+00
 6.00000E+00
 9.00000E+00
 8.00000E+00
 3.00000E+00
 2.00000E+00
 7.00000E+00
 1.00000E+00
```

Example 2

This example shows the vector file for the same geometry.

```
These are the velocity values for a small geometry
part
         1
coordinates
block
 1.00000E+00
 2.00000E+00
 3.00000E+00
 4.00000E+00
 5.00000E+00
 6.00000E+00
```

```
 7.00000E+00
 1.00000E+00
 2.00000E+00
 3.00000E+00
 4.00000E+00
 5.00000E+00
 6.00000E+00
 7.00000E+00
 1.00000E+00
 2.00000E+00
 3.00000E+00
 4.00000E+00
 5.00000E+00
 6.00000E+00
 7.00000E+00
part
         2
block
 4.00000E+00
 6.00000E+00
 9.00000E+00
 8.00000E+00
 3.00000E+00
 2.00000E+00
 7.00000E+00
 1.00000E+00
 6.00000E+00
 9.00000E+00
 3.00000E+00
 3.00000E+00
 4.00000E+00
 6.00000E+00
 9.00000E+00
 8.00000E+00
 3.00000E+00
 2.00000E+00
 7.00000E+00
 1.00000E+00
 6.00000E+00
 9.00000E+00
 3.00000E+00
 3.00000E+00
```

EnSight Gold variable files for per_element variables contain values for each element of designated types of designated Parts. First comes a single description line. Second comes a Part line. Third comes a line containing the part number. Fourth comes an element type line and then comes the value for each element of that type and part. If it is a scalar variable, there is one value per element, while for vector variables there are three values per element. (The number of elements of the given type are obtained from the corresponding EnSight Gold geometry file.)

C Binary form:

SCALAR FILE:

```
description line 1                                    80 chars
part                                                  80 chars
#                                                      1 int
element type                                          80 chars
```

```
s_e1 s_e2 ... s_ne                                      ne floats
element type                                            80 chars
 .
 .
part                                                    80 chars
 .
 .
part                                                    80 chars
#                                                        1 int
block                    # mm = (i-1)*(j-1)*(k-1)       80 chars
s_m1 s_m2 ... s_mm                                      mm floats
```

VECTOR FILE:

```
description line 1                                      80 chars
part                                                    80 chars
#                                                        1 int
element type                                            80 chars
vx_e1 vx_e2 ... vx_ne                                   ne floats
vy_e1 vy_e2 ... vy_ne                                   ne floats
vz_e1 vz_e2 ... vz_ne                                   ne floats
element type                                            80 chars
 .
 .
part                                                    80 chars
 .
 .
part                                                    80 chars
#                                                        1 int
block                    # mm = (i-1)*(j-1)*(k-1)       80 chars
vx_m1 vx_m2 ... vx_mm                                   mm floats
vy_m1 vy_m2 ... vy_mm                                   mm floats
vz_m1 vz_m2 ... vz_mm                                   mm floats
```

Fortran Binary form:

SCALAR FILE:

```
'description line 1'                                    80 chars
'part'                                                  80 chars
'#'                                                      1 int
'element type'                                          80 chars
's_e1 s_e2 ... s_ne'                                    ne floats
'element type'                                          80 chars
 .
 .
'part'                                                  80 chars
 .
 .
'part'                                                  80 chars
'#'                                                      1 int
'block'                  # mm = (i-1)*(j-1)*(k-1)       80 chars
's_m1 s_m2 ... s_mm'                                    mm floats
```

VECTOR FILE:

```
‘description line 1’                                    80 chars
‘part’                                                  80 chars
‘#’                                                      1 int
‘element type‘                                          80 chars
‘vx_e1 vx_e2 ... vx_ne’                                 ne floats
‘vy_e1 vy_e2 ... vy_ne’                                 ne floats
‘vz_e1 vz_e2 ... vz_ne’                                 ne floats
‘element type’                                          80 chars
 .
 .
‘part’                                                  80 chars
 .
 .
‘part’                                                  80 chars
‘#’                                                      1 int
‘block’                 # mm = (i-1)*(j-1)*(k-1)        80 chars
‘vx_m1 vx_m2 ... vx_mm’                                 mm floats
‘vy_m1 vy_m2 ... vy_mm’                                 mm floats
‘vz_m1 vz_m2 ... vz_mm’                                 mm floats
```

ASCII form:

SCALAR FILE:

```
description line 1                                      A (max of 80 typ)
part                                                    A
#                                                       I10
element type                                            A
s_e1                                                    12.5  1/line (ne)
s_e2
 .
 .
s_ne
element type                                            A
 .
 .
part                                                    A
 .
 .
part                                                    A
#                                                       I10
block                   # mm = (i-1)*(j-1)*(k-1)        A
s_m1                                                    E12.5  1/line (mm)
s_m2
 .
 .
s_mm
```

VECTOR FILE:

```
description line 1                                      A (max of 80 typ)
part                                                    A
#                                                       I10
element type                                            A
vx_e1                                                   E12.5  1/line (ne)
vx_e2
 .
```

```
 .
vx_ne
vy_e1                                          E12.5  1/line (ne)
vy_e2
 .
 .
vy_ne
vz_e1                                          E12.5  1/line (ne)
vz_e2
 .
 .
vz_ne
element type                                   A
 .
 .
part                                           A
 .
 .
part                                           A
#                                              I10
block                 # mm = (i-1)*(j-1)*(k-1)    A
vx_m1                                          E12.5  1/line (mm)
vx_m2
 .
 .
vx_mm
vy_m1                                          E12.5  1/line (mm)
vy_m2
 .
 .
vy_mm
vz_m1                                          E12.5  1/line (mm)
vz_m2
 .
 .
vz_mm
```

Per element Variable:

Example 1

This example shows a per_element scalar file for a single Part geometry with eight triangles and 9 quads and block data for a 2x2x2 structured Part.

```
These are the temperature values at elements for a small geometry.
part
         1
tria3
 1.00000E+00
 1.10000E+00
 1.20000E+00
 1.30000E+00
 1.30000E+00
 1.20000E+00
 1.10000E+00
 1.00000E+00
quad4
 4.00000E+00
 6.00000E+00
 9.00000E+00
 8.00000E+00
 3.00000E+00
```

```
 2.00000E+00
 7.00000E+00
 1.00000E+00
 5.00000E+00
part
         1
block
 5.00000E+00
```

Example 2

This example shows a vector file for a similar model. The only difference is that the triangles and quads have been placed in different Parts and there is no structured Part data.

```
These are the velocity values at elements of a small model.
part
         1
tria3
 1.00000E+00
 1.10000E+00
 1.20000E+00
 1.30000E+00
 1.30000E+00
 1.20000E+00
 1.10000E+00
 1.00000E+00
 1.00000E+00
 1.10000E+00
 1.20000E+00
 1.30000E+00
 1.30000E+00
 1.20000E+00
 1.10000E+00
 1.00000E+00
 1.00000E+00
 1.10000E+00
 1.20000E+00
 1.30000E+00
 1.30000E+00
 1.20000E+00
 1.10000E+00
 1.00000E+00
part
         2
quad4
 4.00000E+00
 8.00000E+00
 6.00000E+00
 3.00000E+00
 9.00000E+00
 8.00000E+00
 4.00000E+00
 2.00000E+00
 6.00000E+00
 4.00000E+00
 8.00000E+00
 6.00000E+00
 3.00000E+00
 9.00000E+00
 8.00000E+00
 4.00000E+00
 2.00000E+00
```

```
6.00000E+00
4.00000E+00
8.00000E+00
6.00000E+00
3.00000E+00
9.00000E+00
8.00000E+00
4.00000E+00
2.00000E+00
6.00000E+00
```

EnSight Gold Measured/Particle File Format

The format of a Measured/Particle geometry file is identical to that of the EnSight6 data format.

EnSight5 Format Data Files

Note: The EnSight6 format replaces and includes all aspects of the older EnSight5 format. This description is included for completeness but use of the EnSight6 format with EnSight 6.x and later versions is encouraged!

EnSight5 data consists of the following files:

- Geometry (required)
- Results (optional) (points to other variable files and possibly to changing geometry files)
- Measured (optional) (points to measured geometry and variable files)

The results file contains information concerning scalar and vector variables. EnSight makes no assumptions regarding the physical significance of the scalar and vector variables. These files can be from any discipline. For example, the scalar file can include such things as pressure, temperature, and stress. The vector file can be velocity, displacement, or any other vector data.

All variable results for EnSight5 are contained in disk files—one variable per file. Additionally, if there are multiple time steps, there must be a set of disk files for each time step.

Sources of EnSight5 data include the following:

- Data that can be translated to conform to the EnSight5 data format
- Data that originates from one of the translators supplied with the EnSight application

The EnSight5 format supports a defined element set as shown below. The data must be defined in this element set. Elements that do not conform to this set must either be subdivided or discarded.

Supported EnSight5 Elements

The elements that are supported by the EnSight5 format are:

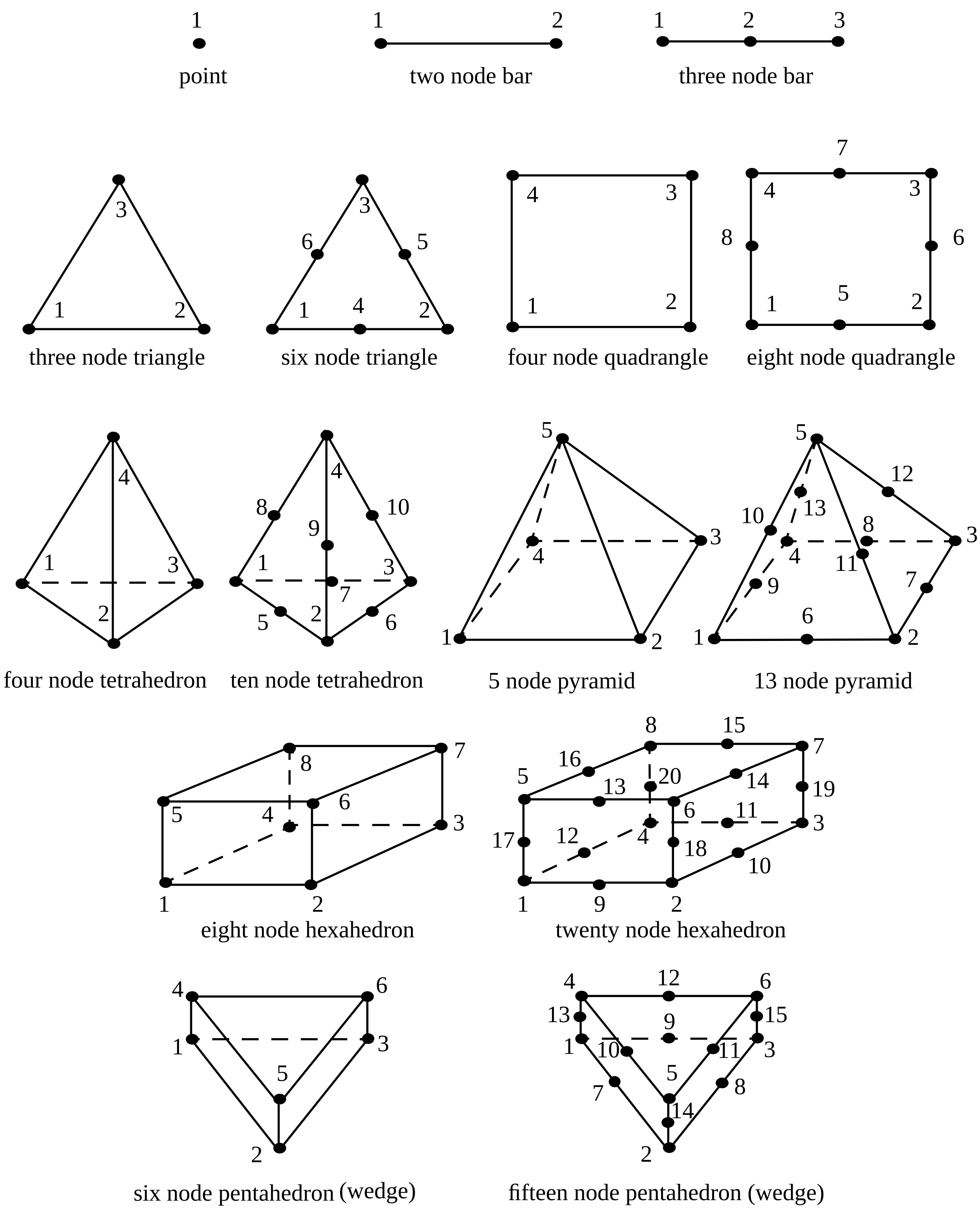

Figure 2-24
Supported EnSight5 Elements

EnSight5 Geometry File Format

The EnSight5 format consists of keywords followed by information. The following items are important to remember when working with EnSight5 geometry files:

1. You do not have to assign node IDs. If you do, the element connectivities are based on the node numbers. If you let EnSight assign the node IDs, the nodes are considered to be sequential starting at node 1, and element connectivity is done accordingly. If node IDs are set to off, they are numbered internally; however, you will not be able to display or query on them. If you have node IDs in your data, you can have EnSight ignore them by specifying "node id ignore." Using this option may reduce some of the memory taken up by the Client and Server, but remember that display and query on the nodes will not be available.
2. You do not need to specify element IDs. If you specify element IDs, or you let EnSight assign them, you can show them on the screen. If they are set to off, you will not be able to show or query on them. If you have element IDs in your data you can have EnSight ignore them by specifying "element id ignore." Using this option will reduce some of the memory taken up by the Client and Server. This may or may not be a significant amount, and remember that display and query on the elements will not be available.
3. The format of integers and real numbers **must be followed** (See the Geometry Example below).
4. Integers are written out using the following integer format:

From C:	`8d` format
From FORTRAN:	`i8` format

   Real numbers are written out using the following floating-point format:

From C:	`12.5e` format
From FORTRAN:	`e12.5` format

   The number of integers or reals per line must also be followed!
5. By default, a Part is processed to show the outside boundaries. This representation is loaded to the Client host system when the geometry file is read (unless other attributes have been set on the workstation, such as feature angle).
6. Coordinates must be defined before any Parts can be defined. The different elements can be defined in any order (that is, you can define a hexa8 before a bar2).

Generic Format

Not all of the lines included in the following generic example file are necessary:

```
description line 1
description line 2
node id <off/given/assign/ignore>
element id <off/given/assign/ignore>
coordinates
# of points
id x y z
id x y z
```

```
id x y z
.
.
.
part #
description line
point
number of points
id nd nd
id nd nd
id nd nd
.
.
.
bar2
number of bar2's
id nd nd
id nd nd
id nd nd
.
.
.
bar3
number of bar3's
id nd nd nd
id nd nd nd
id nd nd nd
.
.
.
tria3
number of three node triangles
id nd nd nd
id nd nd nd
id nd nd nd
.
.
.
tria6
number of six node triangles
id nd nd nd nd nd nd
.
.
.
quad4
number of quad 4's
id nd nd nd nd
id nd nd nd nd
id nd nd nd nd
id nd nd nd nd
.
.
.
quad8
number of quad 8's
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
.
.
```

```
.
tetra4
number of 4 node tetrahedrons
id nd nd nd nd
id nd nd nd nd
id nd nd nd nd
id nd nd nd nd
.
.
.
tetra10
number of 10 node tetrahedrons
id nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd
.
.
.
pyramid5
number of 5 node pyramids
id nd nd nd nd nd
id nd nd nd nd nd
id nd nd nd nd nd
id nd nd nd nd nd
.
.
.
pyramid13
number of 13 node pyramids
id nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd
.
.
.
hexa8
number of 8 node hexahedrons
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd
.
.
.
hexa20
number of 20 node hexahedrons
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
.
.
.
penta6
number of 6 node pentahedrons
id nd nd nd nd nd nd
id nd nd nd nd nd nd
id nd nd nd nd nd nd
```

```
id nd nd nd nd nd nd
id nd nd nd nd nd nd
.
.
.
penta15
number of 15 node pentahedrons
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
id nd nd nd nd nd nd nd nd nd nd nd nd nd nd nd
.
.
.
```

EnSight5 Geometry Example

The following is an example of an EnSight geometry file:

```
this is an example problem
this is the second description line
node id given
element id given
coordinates
      10
       5 1.00000e+00 0.00000e+00 0.00000e+00
     100 0.00000e+00 1.00000e+00 0.00000e+00
     200 0.00000e+00 0.00000e+00 1.00000e+00
      40 1.00000e+00 1.00000e+00 0.00000e+00
      22 1.00000e+00 0.00000e+00 1.00000e+00
    1000 2.00000e+00 0.00000e+00 0.00000e+00
      55 0.00000e+00 2.00000e+00 0.00000e+00
      44 0.00000e+00 0.00000e+00 2.00000e+00
     202 2.00000e+00 2.00000e+00 0.00000e+00
     101 2.00000e+00 0.00000e+00 2.00000e+00
part 1
This is Part 1, a pretty strange Part
tria3
              2
            101     100     200      40
            201     101       5    1000
tetra4
              1
            102     100     202     101    1000
part 2
This is Part 2, it’s pretty strange also
bar2
              1
            103     101    1000
```

EnSight5 Result File Format

The Result file is an ASCII free format file that contains variable and time step information that pertains to a Particular geometry file. The following information is included in this file:

- Number of scalar variables
- Number of vector variables
- Number of time steps
- Starting file number extension and skip-by value
- Flag that specifies whether there is changing geometry
- Names of the files that contain the values of scalar and vector variables
- The names of the geometry files that will be used for the changing geometry.

The format of the EnSight5 result file is as follows:

- Line 1

  Contains the number of scalar variables, the number of vector variables and a geometry-changing flag. (If the geometry-changing flag is 0, the geometry of the model does not change over time. If it is 1, then there is connectivity changing geometry. If it is 2, then there is coordinate only changing geometry.)

- Line 2

  Indicates the number of time steps that are available.

- Line 3

  Lists the time that is associated with each time step. There must be the same number of values as are indicated in Line 2. This “line” can actually span several lines in the file. You do not have to have one very long line.

- Line 4

  Specified only if more than one time step is indicated in Line 2. The two values on this line indicate the file extension value for the first time step and the offset between files. If the values on this line are 0 5, the first time step available has a subscript of 0, the second time step available has a subscript of 5, the third time step has a subscript of 10, and so on.

- Line 5

  Contains the names of the geometry files that will be used for changing geometry. This line exists only if the flag on Line 1 is set to 1 or 2. The geometry file name must follow the EnSight5 wild card specification.

- Line 6 through Line [5+N] where N is the number of scalar variables specified in Line 1.

  List **BOTH** the file names **AND** variable description that correspond to each scalar variable. There must be a file name for each scalar variable that is specified in Line 1.

If there is more than one time step, the file name must follow the EnSight5 wild card specification. See Note below.

- Lines that follow the scalar variable files.

  List the file names that correspond to each vector variable. There must be a file name for each vector variable that is specified in Line 1. If there is more than one time step, the file name must follow the EnSight5 wild card specification. See Note below.

Note *The variable description is limited to 19 characters in the current release. Variable names must not contain any of the following reserved characters:*

(	[	+	@	!	*	$	:
)	]	-	space	#	^	/	?

The generic format of a result file is as follows:

```
#_of_scalars #_of_vectors geom_chang_flag
#_of_timesteps
time1 time2 time3 .....
start_file_# skip_by_value
geometry_file_name.geo**
scalar0_file_name**   description (19 characters max)
scalar1_file_name**   description
.
.
.
vector0_file_name**   description (19 characters max)
vector1_file_name**   description
.
```

EnSight5 Result File Example 1

The following example illustrates a result file specified for a non-changing geometry file with only one time step:

```
2 1 0
1
0.0
exone.scl0 pressure
exone.scl1 temperature
exone.dis0 velocity
```

EnSight5 Result File Example 2

This example illustrates a result file that specifies a connectivity changing geometry that has multiple time steps.

```
1 2 1
4
1.0 2.0 2.5 5.0
0 1
extwo.geom**
pres.scl** pressure
vel.dis** velocity
grad.dis** gradient
```

The following files would be needed for example 2:

```
extwo.geom00 pres.scl00 vel.dis00 grad.dis00
extwo.geom01 pres.scl01 vel.dis01 grad.dis01
extwo.geom02 pres.scl02 vel.dis02 grad.dis02
extwo.geom03 pres.scl03 vel.dis03 grad.dis03
```

EnSight5 Wild Card Name Specification

If multiple time steps are involved, the file names must conform to the EnSight5 wild-card specification. This specification is as follows:

- File names must include numbers that are in ascending order from beginning to end.
- Numbers in the files names must be zero filled if there is more than one significant digit.
- Numbers can be anywhere in the file name.
- When the file name is specified in the EnSight5 result file, you must replace the numbers in the file with an asterisk(*). The number of asterisks specified is the number of significant digits. The asterisk must occupy the same place as the numbers in the file names.

EnSight5 Variable File Format

Variables files have one description line followed by a value for each node. For a scalar file there is one value per node, while for vector files there are three values per node.

The values **must be written** in the following floating point format (**6 per line** as shown in the examples below):

From C: `12.5e` format

From FORTRAN: `e12.5` format

The format of a variables file is as follows:

- Line 1

  This line is a description line.

- Line 2 through the end of the file contains the values at each node in the model. A generic example:

```
A description line
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
 *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+** *.*****E+**
```

EnSight5 Variable File Example 1

This example shows a scalar file for a geometry with seven defined nodes.

```
These are the pressure values for a 7 node geometry
 1.00000E+00 2.00000E+00 3.00000E+00 4.00000E+00 5.00000E+00 6.00000E+00
 7.00000E+00
```

EnSight5 Variable File Example 2

This example shows the vector file for a geometry with seven defined nodes.

```
These are the velocity values for a 7 node geometry
 1.00000E+00 1.00000E+00 1.00000E+00 2.00000E+00 2.00000E+00 2.00000E+00
 3.00000E+00 3.00000E+00 3.00000E+00 4.00000E+00 4.00000E+00 4.00000E+00
 5.00000E+00 5.00000E+00 5.00000E+00 6.00000E+00 6.00000E+00 6.00000E+00
 7.00000E+00 7.00000E+00 7.00000E+00
```

EnSight5 Measured/Particle File Format

This file allows you to define Particle locations, sizes, etc. to display with the geometry. Typical uses are fuel droplets for combustion analysis or data derived from experiments on prototypes.

The measured/Particle files consist of the following:

- Measured/Particle geometry file
- Measured/Particle results file
- Measured/Particle variables file

The format of the EnSight5 Measured/Particle geometry file is described below.

Note that there is only one description line and there *must* be an ID for each measured point.

Note also that the number of Particles can be different in each of the geometry file (if you have transient data), however, the number of values in each of the corresponding variable files must coincide, and the IDs of the Particles must be consistent in order to track the Particles at intermediate times or locations.

The format of an EnSight5 Measured/Particle geometry file is as follows:

- Line 1

  This line is a description line.

- Line 2

  Indicates that this file contains Particle coordinates. The words “particle coordinates” should be entered on this line without the quotes.

- Line 3

  Specifies the number of Particles.

- Line 4 through the end of the file.

  Each line contains the ID and the X, Y, and Z coordinates of each Particle. The format of this line is “integer real real real” written out in the following format:

  From C: `%8d%12.5e%12.5e%12.5e` format

  From FORTRAN: `i8, 3e12.5` format

A generic measured/Particle geometry file is as follows:

```
A description line
particle coordinates
#_of_Particles
id xcoord ycoord zcoord
id xcoord ycoord zcoord
id xcoord ycoord zcoord
 .
 .
 .
```

EnSight5 Measured Geometry/Particle File Example

The following illustrates an EnSight5 Measured Geometry/Particle file with seven points:

```
This is a simple ensight5 measured geometry/particle file
particle coordinates
       7
     101 0.00000E+00 0.00000E+00 0.00000E+00
     102 1.00000E+00 0.00000E+00 0.00000E+00
     103 1.00000E+00 1.00000E+00 0.00000E+00
     104 0.00000E+00 1.00000E+00 0.00000E+00
     205 5.00000E-01 0.00000E+00 2.00000E+00
     206 5.00000E-01 1.00000E+00 2.00000E+00
     307 0.00000E+00 0.00000E+00-1.50000E+00
```

EnSight5 Measured/ Particle File Format

The format of the EnSight5 Measured/Particle results file is as follows:

- Line 1

  Contains the number of scalar variables, the number of vector variables, and a measured geometry changing flag. If the measured geometry changing flag is 0, only one time step is indicated.

- Line 2

  Indicates the number of available time steps.

- Line 3

  Lists the time that is associated with each time step. The time step information does not have to coincide with the model time step information. This “line” can actually span several lines in the file. You do not have to have one very long line.

- Line 4

  Specified only if Line 2 specifies more than one time step. The line contains two values; the first value indicates the file extension value for the first time step, and the second value indicates the offset between files. If this line contains the values 0 and 5, the first time step has a subscript of 0, the second of 5, the third of 10, and so on.

- Line 5

  Contains the name of the measured geometry file. If there is more than one time step, the file name must follow the EnSight wild card specification.

- Line 6 through Line [5+N] where N is the number of scalar variables specified in Line 1.

  List the file names that correspond to each scalar variable. There must be a file name for each scalar variable that is specified in Line 1. If there is more than one time step, the file name must follow the EnSight wild card specification.

- Lines that follow the scalar variable files.

  List the names of the files that correspond to each vector variable. There must be a file name for each vector variable that is specified in Line 1. If there is more than one time step, the file name must follow the EnSight wild card specification.

A generic EnSight5 Measured/Particle results file is as follows:

```
#_of_scalars #_of_vectors geom_chang_flag
#_of_timesteps
time1 time2 time3 .....
start_file_# skip_by_value
measured_geom_file_name**
scalar0_file_name**   description
scalar1_file_name**   description
.
.
.
vector0_file_name**   description
vector1_file_name**   description
.
.
.
```

Measured/Particle Results File Example 1

This example illustrates an EnSight5 Measured/Particle result file that specifies a non-changing geometry with only one time step:

```
2 1 0
1
0.0
exone.geom
exone.scl0 pressure
exone.scl1 temperature
exone.dis0 velocity
```

Measured/Particle Results File Example 2

This example illustrates an EnSight5 Measured/Particle result file that specifies a changing geometry with multiple time steps:

```
1 2 1
4
1.0 2.0 2.5 5.0
0 1
extwo.geom**
pres.scl** pressure
vel.dis** velocity
grad.dis** gradient
```

The following files are needed for Example 2:

```
extwo.geom00pres.scl00vel.dis00   grad.dis00
extwo.geom01pres.scl01vel.dis01   grad.dis01
extwo.geom02pres.scl02vel.dis02   grad.dis02
extwo.geom03pres.scl03vel.dis03   grad.dis03
```

Measured /Particle Results Variable files

The EnSight5 Measured/Particle variable files referred to in the measured Results file follow the same format as EnSight5 Variable files. The number of values in each of these variable files must correspond properly to the number of Particles in the corresponding measured geometry files.

Writing EnSight5 Binary Files

This section describes the EnSight5 binary files. This format is used to increase the speed of reading data into EnSight. A utility exists for converting EnSight5 ASCII files to EnSight5 binary files—it is called asciitobin5 and is found on the release tape under ensight/server/utilities/asciitobin5.

For binary files, there is a header that designates the type of binary file. This header is: “C Binary” or “Fortran Binary.” This must be the first thing in the file. The format for the file is then essentially the same format as the ASCII format, with the following exceptions:

> The ASCII format puts the node and element ids on the same “line” as the corresponding coordinates. The BINARY format writes all node id’s then all coordinates.
>
> The ASCII format puts all element id’s of a type within a Part on the same “line” as the corresponding connectivity. The BINARY format writes all the element ids for that type, then all the corresponding connectivities of the elements.

In all the descriptions of binary files that follow, the number on the left end of the line corresponds to the type of write of that line, according to the following code:

1. This is a write of 80 characters to the file:

   C example:
   ```
   char buffer[80];
   strcpy(buffer,”C Binary”);
   fwrite(buffer,sizeof(char),80,file_ptr);
   ```

   FORTRAN:
   ```
   character*80   buffer
   buffer = “Fortran Binary”
   write(10) buffer
   ```

2. This is a write of a single integer:

   C example: `fwrite(&num_nodes,sizeof(int),1,file_ptr);`

   FORTRAN: `write(10) num_nodes`

3. This is a write of an integer array:

   C example: `fwrite(node_ids,sizeof(int),num_nodes,file_ptr);`

   FORTRAN: `write(10) (node_ids(i),i=1,num_nodes)`

4. This is a write of a float array:

   C example: `fwrite(coords,sizeof(float),3*num_nodes,file_ptr);`

   FORTRAN: `write(10) ((coords(i,j),i=1,3),j=1,num_nodes)`

(Note: Coords is a single precision array, double precision will not work!)

EnSight5 Binary Geometry File Format

An EnSight5 binary geometry file contains information in the following order:

(1) <C Binary/Fortran Binary>

(1) description line 1
(1) description line 2
(1) node id <given/off/assign/ignore>
(1) element id <given/off/assign/ignore>
(1) coordinates
(2) #_of_points
(3) [point_ids]
(4) coordinate_array
(1) Part #
(1) description line
(1) element_type
(2) #_of_element_type
(3) [element_ids] for the element_type
(3) connnectivities for the element_type
(1) element_type
(2) #_of_element_type
(3) [element_ids] for the element_type
(3) connnectivities for the element_type
.
.
.
(1) Part #
(1) description line
(1) element_type
(2) #_of_element_type
(3) [element_ids] for the element_type
(3) connnectivities for the element_type
(1) element_type
(2) #_of_element_type
(3) [element_ids] for the element_type
(3) connnectivities for the element_type
.
.
.

Binary Scalar

An EnSight5 binary scalar file contains information in the following order:

(1) description line
(4) scalar_array

Binary Vector

An EnSight5 binary vector file contains information in the following order:

(1) description line
(4) vector_array

Binary Measured

An EnSight5 binary measured/Particle geometry file contains information in the following order:

(1) <C Binary/Fortran Binary>

(1) description line 1
(1) particle coordinates
(2) #_of_points
(3) point_ids
(4) coordinate_array

FLUENT Data File Format

This section describes the FLUENT results file format and provides an example of this file. The FLUENT result file is a slightly modified EnSight results file and provides a way to describe multiple time-step FLUENT Universal files to EnSight.

When using multiple FLUENT files with this result file definition, you *must* make sure that the files contain the same defined variables. In other words, any variable that exists in one must exist in all.

FLUENT Data Files

FLUENT input data consists of the following:

- Universal file (required)
- Results file (optional)
- EnSight Measured/Particle files (optional)

FLUENT Result file format

The Result file contains time step and universal file information for each available time step. You are *not* required to supply this file unless you have multiple time steps. The following information is included in this file:

- Number of time steps
- Starting file number extension and skip-by value
- Simulation Time Values
- Name of the universal file with EnSight wild card specification.

The format of the Result file is as follows:

- Line 1

  Indicates the number of time steps that are available.

- Line 2

  Lists the time that is associated with each time step. There must be the same number of values as are indicated in Line 1. This “line” can actually span several lines in the file. You don’t have to have one very long line.

- Line 3

  The two values on this line indicate the file extension value for the first time step and the offset between files. If the values on this line are 0 5, the first time step available has a subscript of 0, the second time step available has a subscript of 5, the third time step has a subscript of 10, and so on.

- Line 4

  Contains the name of the universal file that will be used for changing geometry. The geometry file name must follow the EnSight wild card specification.

-

FLUENT Format

The generic format of a result file is as follows:

```
#_of_timesteps
time1 time2 time3 .....
start_file_# skip_by_value
universal_file_name***.geo
```

FLUENT Example

This example illustrates a FLUENT result file:

```
4
1.0 2.0 2.5 5.0
0 1
extwo**.uni
```

The following FLUENT universal files will need to exist for the result file:

```
extwo00.uni
extwo01.uni
extwo02.uni
extwo03.uni
```

Movie.BYU Data File Format

For transient cases, you must supply an EnSight result file. The result file for the Movie.BYU case is exactly the same as for EnSight5.

PLOT3D Data File Format

PLOT3D input data consists of the following:

- Geometry file (required) (GRID file).
- Results file (optional).
- EnSight Measured/Particle files (optional).

PLOT3D data files can be read as:

Workstation: ASCII, C Binary, or FORTRAN binary

Cray: ASCII, C Binary, or COS-Blocked FORTRAN binary

Due to the different of number representations on a Cray Research vector system and workstations, binary files created on a Cray Research vector system can *not* be read on the workstation, and visa versa.

EnSight attempts to ensure that the format of the file being read matches the format you have selected in the Data Reader dialog. However, if you specify that the file is C binary, and it is really FORTRAN binary, this will not be detected and erroneous values will be loaded.

EnSight reads the geometry (xyz files) directly. However, an EnSight-like results file (described below) is needed in order to read the results, unless a “standard” Q-file is provided in its place.

PLOT3D Geometry file notes

The following information is required in order to read PLOT3D files correctly:

1. whether there is Iblanking information in the file
2. whether files are in ASCII, C Binary, or FORTRAN binary
3. whether the file is “Single Zone” or Multi-Zoned”
4. whether the model is 1D, 2D, or 3D in nature.

Iblanking can be one of the following:

0 = Outside (Blanked Out)

1 = Inside

2 = Interior boundaries

<0 = zone that neighbors

If single zone with Iblanking, you can build EnSight Parts from the inside portions, blanked-out portions, or internal boundary portions. If single zone, you can also specify I, J, K limiting ranges for Parts to be created.

If Multi-zoned with Iblanking, you can additionally build Parts that are the boundary between two zones. (For boundary you must specify exactly two zones.)

If Multi-zoned and not using the “between boundary” option, a Part can span several zones.

If Multi-zoned, the dimension of the problem is forced to be 3D.

There can be nodes in different zones which have the same coordinates. No attempt has been made to merge these. Thus, on shared zone boundaries, there will likely be nodes on top of nodes. One negative effect of this is that node labels will be on top of each other.

Currently EnSight only prints out the global conditions in the solution file, fsmach, alpha, re, and time. It does not do anything else with them.

Node and element numbers are assigned in a sequential manner. Queries can be made on these node and element numbers or on nodes by I, J, and K.

PLOT3D Result file format

The PLOT3D result file was defined by CEI and is very similar to the EnSight results file and contains information needed to relate variable names to variable files, step information, etc. There is a slight variation from the normal EnSight results file because of the differences between the solution (Q file) and function files. The difference lies on the lines which relate variable filenames to a description. These lines have the following format:

```
<filename> <type> <number(s)> <description>
```

See PLOT3D Result File below for the definition of each.

The following information is included in a PLOT3D result file:

- Number of scalar variables
- Number of vector variables
- Number of time steps
- Starting file number extension and skip-by value
- Flag that specifies whether there is changing geometry.
- Names of the files that contain the values of scalar and vector variables. An indication as to the type of the file being used for the variable, which variable in the file and the name given to that variable.
- The names of the geometry files that will be used for the changing geometry.

Generic PLOT3D Result File Format

The format of the Result file is as follows:

- Line 1

  Contains the number of scalar variables, the number of vector variables and a geometry changing flag. If the geometry changing flag is 0, the geometry of the model does not change over time. Only the coordinates can change for a PLOT3D file at present time.

- Line 2

  Indicates the number of time steps that are available.

- Line 3

Lists the time that is associated with each time step. There must be the same number of values as are indicated in Line 2. This “line” can actually span several lines in the file.

- Line 4

  Specified only if more than one time step is indicated in Line 2. The two values on this line indicate the file extension value for the first time step and the offset between files. If the values on this line are 0 5, the first time step available has a subscript of 0, the second time step available has a subscript of 5, the third time step has a subscript of 10, and so on.

- Line 5

  This line exists only if the changing geometry flag on Line 1 has been set to 1. Line contains name of the PLOT3D xyz file. The file name must follow the EnSight wild card specification.

- Line 6 through Line [5+N] where N is the number of scalar variables specified in Line 1.

  List the file names that correspond to each scalar variable. There must be a file name for each scalar variable that is specified in Line 1. If there is more than one time step, the file name must follow the EnSight wild card specification.

  These lines also contain the type of file being used, solution or function, and the location of the variable value in the file. The contents are:

  ```
  <filename> <type> <number> <description>
  ```

  where filename is the name of solution file or function file containing the variable; type is “S” for solution file, or “F” for function file; numberis which variable in the file to use (specify just one number); and description is the Description of the variable.

  The solution file (“s”) is the traditional .q file in which normally the first variable is density, the second through fourth variables are the components of momentum, and the fifth variable is total energy.

- Lines that follow the scalar variable files.

  List the file names that correspond to each vector variable. There must be a file name for each vector variable that is specified in Line 0. If there is more than one time step, the file name must follow the EnSight wild card specification.

  These lines also contain the type of file being used, solution or function, and the location(s) of the variable values in the file. The contents are:

  ```
  <filename> <type> <numbers> <description>
  ```

  where filename is the name of solution file or function file containing the variable; type is “S” for solution file, or “F” for function file; numbersare which variables in the file to use (specify just three numbers); and description is the Description of the variable.

The generic format of the result file is as follows:

```
#_of_scalars #_of_vectors geom_chng_flag
#_of_timesteps
```

```
time1 time2 time3 .....
start_file_# skip_by_value
geometry_file_name.geo**
scalar0_file_name** type # description
scalar1_file_name** type # description
.
.
.
vector0_file_name** type # # # description
vector1_file_name** type # # # description
.
.
.
```

PLOT3D Example

This example illustrates a result file that specifies a non-changing geometry with only one time step.

```
3 2 0
1
0.0
block.sol S 1 Density
block.sol S 5 Total_Energy
block.scl F 1 Temperature
block.var F 1 2 3 Displacement
block.sol S 2 3 4 Momentum
```

Thus, this model will get two scalars from the solution file (block.sol). The first is Density in the first location in the file and the next is Total energy in the fifth location in the solution file. It will also get a Temperature scalar from the first location in the function file (block.scl).

It will get a Displacement vector from the function file called block.var. The three components of this vector are in the 1st, 2nd, and 3rd locations in the file. Finally, a Momentum vector will be obtained from the 2nd, 3rd, and 4th locations of the solution file.

Vectors can be 1D, 2D, or 3D. For a vector, always provide three numbers, but a zero will indicate that a component is empty, thus:

```
block.var F 1 0 3 XZ_Displacement
```

would be a 2D vector variable with components only in the X–Z plane.

Note: *A "standard" Q-file can be substituted for PLOT3D result file format if desired. A"standard" Q-file has 5 variable components (First is density, then the three components of momentum, and last is energy).*

FAST UNSTRUCTURED Data File Format

FAST UNSTRUCTURED input data consists of the following:

- Geometry file (required) (GRID file).
- Results file (optional).
- EnSight Measured/Particle files (optional).

FAST UNSTRUCTURED data files can be read as:

Workstation: ASCII, C Binary, or FORTRAN binary

Cray: ASCII, C Binary, or COS-Blocked FORTRAN binary

Due to the different number of representations on a Cray Research vector system and workstations, binary files created on a Cray Research vector system can *not* be read on the workstation, and visa versa.

EnSight reads the geometry (grid files) directly. However, an EnSight-like results file is needed in order to read the results unless a “standard” Q-file is provided in its place. See FAST UNSTRUCTURED Result File below.

FAST UNSTRUCTURED Geometry file notes

Only the single zone format can be read into EnSight. Any tetrahedral elements will be placed into the first “domain” Part. Triangular elements are placed into Parts based on their “tag” value.

The FAST UNSTRUCTURED solution file or function file formats can be used for variable results. The I J K values need to be I=Number of points and J=K=1. This does require the use of a modified EnSight results file as explained below.

Node and element numbers are assigned sequentially allowing for queries to be made within EnSight. Tetrahedron elements will be assigned before triangular elements.

FAST UNSTRUCTURED Result file format

The FAST UNSTRUCTURED result file was defined by CEI and is very similar to the EnSight results file and contains information needed to relate variable names to variable files, step information, etc. There is a slight variation from the normal EnSight results file because of the differences between the solution (Q file) and function files. The difference lies on the lines which relate variable filenames to a description. These lines have the following format:

```
<filename> <type> <number(s)> <description>
```

See FAST UNSTRUCTURED Result File below for the definition of each.

The following information is included in a FAST UNSTRUCTURED result file:

- Number of scalar variables
- Number of vector variables
- Number of time steps
- Starting file number extension and skip-by value
- Flag that specifies whether there is changing geometry.

- Names of the files that contain the values of scalar and vector variables. An indication as to the type of the file being used for the variable, which variable in the file and the name given to that variable.
- The names of the geometry files that will be used for the changing geometry.

Generic FAST UNSTRUCTURED Result File Format

The format of the Result file is as follows:

- Line 1

  Contains the number of scalar variables, the number of vector variables and a geometry changing flag. If the geometry changing flag is 0, the geometry of the model does not change over time. If the flag is 1, the geometry can change connectivity. If the flag is 2, only coordinates can change.

- Line 2

  Indicates the number of time steps that are available. If this number is positive, then line 3 information must be present. If this number is negative, then Line 3 information must not be present and the times will be read from the solution file. Thus, one must have a solution file in one of the lines from Line 6 on.

- Line 3

  Lists the time that is associated with each time step. There must be the same number of values as are indicated in Line 2. This “line” can actually span several lines in the file. Specify only if Line 2 value is positive.

- Line 4

  Specified only if more than one time step is indicated in Line 2. The two values on this line indicate the file extension value for the first time step and the offset between files. If the values on this line are 0 5, the first time step available has a subscript of 0, the second time step available has a subscript of 5, the third time step has a subscript of 10, and so on.

- Line 5

  This line exists only if the changing geometry flag on Line 1 has been set to 1 or 2. Line contains name of the FAST UNSTRUCTURED grid file. The file name must follow the EnSight wild card specification.

- Line 6 through Line [5+N] where N is the number of scalar variables specified in Line 1.

  List the file names that correspond to each scalar variable. There must be a file name for each scalar variable that is specified in Line 1. If there is more than one time step, the file name must follow the EnSight wild card specification.

  These lines also contain the type of file being used, solution or function, and the location of the variable value in the file. The contents are:

  ```
  <filename> <type> <number> <description>
  ```

  where filename is the name of solution file or function file containing the variable; type is “S” for solution file, or “F” for function file; numberis which variable in the file to use (specify just one number); and `description` is the Description of the variable.

The solution file (“s”) is the traditional .q file in which normally the first variable is density, the second through fourth variables are the components of momentum, and the fifth variable is total energy.

- Lines that follow the scalar variable files.

  List the file names that correspond to each vector variable. There must be a file name for each vector variable that is specified in Line 0. If there is more than one time step, the file name must follow the EnSight wild card specification.

  These lines also contain the type of file being used, solution or function, and the location(s) of the variable values in the file. The contents are:

  ```
  <filename> <type> <numbers> <description>
  ```

  where filename is the name of solution file or function file containing the variable; type is “S” for solution file, or “F” for function file; numbersare which variables in the file to use (specify just three numbers); and description is the Description of the variable.

The generic format of the result file is as follows:

```
#_of_scalars #_of_vectors geom_chng_flag
#_of_timesteps
time1 time2 time3 .....
start_file_# skip_by_value
geometry_file_name.geo**
scalar0_file_name** type # description
scalar1_file_name** type # description
.
.
.
vector0_file_name** type # # # description
vector1_file_name** type # # # description
.
.
.
```

FAST UNSTRUCTURED Example

This example illustrates a result file that specifies a non-changing geometry with only one time step.

```
3 2 0
1
0.0
block.sol S 1 Density
block.sol S 5 Total_Energy
block.scl F 1 Temperature
block.var F 1 2 3 Displacement
block.sol S 2 3 4 Momentum
```

Thus, this model will get two scalars from the solution file (block.sol). The first is Density in the first location in the file and the next is Total energy in the fifth location in the solution file. It will also get a Temperature scalar from the first location in the function file (block.scl).

It will get a Displacement vector from the function file called block.var. The three components of this vector are in the 1st, 2nd, and 3rd locations in the file. Finally, a Momentum vector will be obtained from the 2nd, 3rd, and 4th locations of the solution file.

Example 2 is somewhat similar, except that it is transient, with coordinate changing geometry. Note also that the times will come from the solution file.

```
3 2 2
-10
0 1
block***.grid
block***.sol S 1 Density
block***.sol S 5 Total_Energy
block***.scl F 1 Temperature
block***.var F 1 2 3 Displacement
block***.sol S 2 3 4 Momentum
```

Note: *A "standard" Q-file can be substituted for FAST UNSTRUCTURED result file format if desired. A "standard" Q-file has 5 variable components (First is density, then the three components of momentum, and last is energy).*

User Defined Reader File Format

A user-defined reader capability is included in EnSight which allows otherwise unsupported structured or unstructured data to be read directly. In other words, the user can create their own data reader. The necessary code for a defined library of routines must be created, compiled, and loaded into a dynamic shared library. According to an environment variable, this library will be loaded at run time and effectively adds an internal reader into EnSight.

The process for creating a user-defined reader is explained in detail in a README file on the installation CD or in your installation directory.

On the CD: /CDROM/ensight62/userd_defined/readers/README

In installation directory: $ENSIGHT6_HOME/ensight62/userd_defined/readers/README

Color Palette File Format

This file defines the colors that are used with the EnSight Color Selector. If EnSight does not find a definition file it uses a default palette. If, however, it does find a file (the file must be called ensight.colpal.default and be located in the .ensight6 directory of the users home directory) at start-up it will read your colors and show them in the Color Selector.

The format of the ensight.colpal.default file is as follows:

- Line 1: “Version 6.0”
- Line 2 through Line 77

  Three integers, one for each color (red, green, blue), ranging from

  0 (no intensity) to 255 (full intensity).

Connection Information File Format

EnSight saves a file on the Client host system, called `ensight.connect.default` whenever you connect the Server via the auto connect feature. The next time you start EnSight, it will read this file and display your previous connection information. This file is normally saved to and read from the .ensight6 directory of the users home directory. But, any local file will override this location process.

The complete ASCII text file contains the following Server and/or plotter system information.

```
server
machine SERVER_SYSTEM_ID
executable [SERVER_EXE_PATH/]ensight.server
directory SERVER_WORKING_DIRECTORY
login_id SERVER_LOGIN_ID
```

Each line of the file consists of a descriptive keyword that is usually followed by an appropriate system variable. The system variables are shown above with generic abbreviations in capital letters.

Keyword	Description
`server`	Denotes that following keywords and variables pertain to how the program `ensight6.server` is started via an automatic connection.
`machine`	The id or hostname of the system where the program is executed. This defaults to your Client host system hostname.
`executable`	The complete path to the executable program. This defaults to executing ensight6.server (which must be in your defined UNIX search path). This path is normally defined in your .login or .cshrc file in your home directory (for C shell users).
`directory`	The directory that you wish the Server to execute from on the Server host system. You may want to specify the directory that contains your data files on the Server host system. This defaults to your home directory on the (Server or plotter) host system for a distributed connection. It defaults to the Client's working directory when in standalone mode.
`login_id`	Your alternate login id on the Server host system. This defaults to your Client host system login id. (This option is only applicable to distributed connections).

Function Palette File Format

A function palette file is saved using the Function Editor when you save (one or more) function color palettes. The following is an example function palette file:

```
palette ‘velocity’
variable_type vector
variable ‘velocity’
type continuous
limit_fringes no
scale linear
number_of_levels 5
colors
0.000000 0.000000 1.000000
0.000000 1.000000 1.000000
0.000000 1.000000 0.000000
1.000000 1.000000 0.000000
1.000000 0.000000 0.000000
values
0.100341
0.301022
0.501704
0.702385
0.903067
```

Many lines of the file consists of a descriptive keyword followed by an appropriate value. In other areas the keyword is used to start a block of information. The values are all free format real or integer numbers or string constants. The palette name must have single quotes around each name. The string keywords and constant values must match exactly.

Keyword	Description
`palette`	Name of the palette when one name is present. Name of the subpalette when two names are present (ex. palette ‘velocity’’xcomp’)
`variable`	Name of the variable used with the palette.
`variable_type`	Type of the variable, scalar or vector.
`type`	Type of the palette, continuous or banded.
`limit_fringes`	Indicates if the palette is set up for limiting fringe. If it is, the options are `by_Part` or `by_invisible`.
`scale`	Indicates whether the palette scale is linear, logarithmic, or quadratic.
number_of_levels	Indicates the number of levels defined for the palette.
colors	Indicates the start of a block of RGB triplets, 1 triplet per line. There will be the same number of lines as there are levels.
values	Indicates the start of a block of level values. There will be the same number of values as there are levels.

XY Data Format

This file is saved using the Save section of the Query Entity dialog. The file can contain one or more curves. The following is an example XY Data file:

Line	Contents of Line
1	2
2	Distance vs. Temperature for Line Tool
3	Distance
4	Temperature
5	1
6	5
7	0.0 4.4
8	1.0 5.8
9	2.0 3.6
10	3.0 4.6
11	4.0 4.8
12	Distance vs. Pressure for Line Tool
13	Distance
14	Pressure
15	2
16	4
17	0.00 1.2
18	0.02 1.1
19	0.04 1.15
20	0.06 1.22
21	3
22	1.10 1.30
23	1.12 1.28
24	1.14 1.25

Line 1 contains the (integer) number of curves in the file.

Line 2 contains the name of the curve.

Line 3 contains the name of the X-Axis.

Line 4 contains the name of the Y-Axis.

Line 5 contains the number of curve segments in this curve.

Line 6 contains the number of points in the curve segment.

Lines 7-11 contain the X-Y information.

Line 12 contains the name of the second curve.

Line 13 contains the name of the X-Axis

Line 14 contains the name of the Y-Axis

Line 15 contains the number of curve segments in this curve. (For the second curve, the first segment contains 4 points, the second 3 points.)

Window Position File Format

To save a window position file, click Prefs>Save Window Positions... from the Main Menu. When you select this button, the current position of major dialog windows is saved to the `ensight.winpos.default` file. In general, this file contains dialog position and size information, along with information about the states of the expandable sections of dialogs.

This file is normally saved to and read from the .ensight6 directory of the users home directory. If the file is in the Client working directory it will be read from and saved to that directory instead.

Only major dialogs are affected; the miscellaneous pop-up dialogs are not specified. You do not have to include every dialog and every section listed. EnSight will process the ones provided.

Window File Format The format of the EnSight window position file is as follows:

- Line 1: Font Size

  Integer specifying font size for dialog labels.

- Lines 2 to N: Dialog Title, Size, & Location

  `String: [IntegerXInteger+]Integer+Integer` specifying Dialog title: Width x Height + Xloc + Yloc. The dialog title of each window can be shortened using the * as a meta character. For example, the string title Transformation Editor: 0+815 can be shortened to *Transform*: 0+815. Be careful that your abbreviated name does not match any other names, or the position of all those names will be changed.

- Line N+1: List Separator String

  Character string –`Section Expansion Information`– to separate dialog size and location information from section-open information.

- Lines N+2 to End: Section Expansion Toggles

  `Dialog->Section[->Section]: open|closed` character strings indicating whether corresponding dialog section is open or closed.

The following is an example window position file:

```
fontsize: 13
EnSight: 910x984+369+31
Transformation Editor: 390x381+180+368
Command: 300x0+0+682
Connect Server: 137+0
Query Dataset: 0+0
```

Periodic Match File Format

This is an optional file which can be used in conjunction with models which have rotational or translational computational symmetry (or periodic boundary conditions).

When a model piece is created with periodic boundary conditions, there is usually a built-in correspondence between two faces of the model piece. If you transform a copy of the model piece properly, face 1 of the copy will be at the same location as face 2 of the original piece. It is desirable to know the corresponding nodes between face 1 and face 2 so they will not be duplicated when the transformation takes place. This correspondence of nodes can be provided in a periodic match file as indicated below. (Please note that if a periodic match file is not provided, EnSight will attempt to eliminate any duplicate nodes using a hashing scheme. This scheme has been shown to work quite well, but may not catch all duplicates.)

The transformation type and delta value are contained in the file. The periodic match file is an ASCII free format file. It can be thought of as a series of node pairs, where a node pair is the node number of face 1 and its corresponding node number on face 2.

The first line in the file is either “rotate” or “translate”. The second line in the file contains the rotation angle in degrees or the three translational delta values (dx, dy, dz). The third line in the file contains the `Number_of_node_pairs`, followed by that many, node pairs.

Simple example:

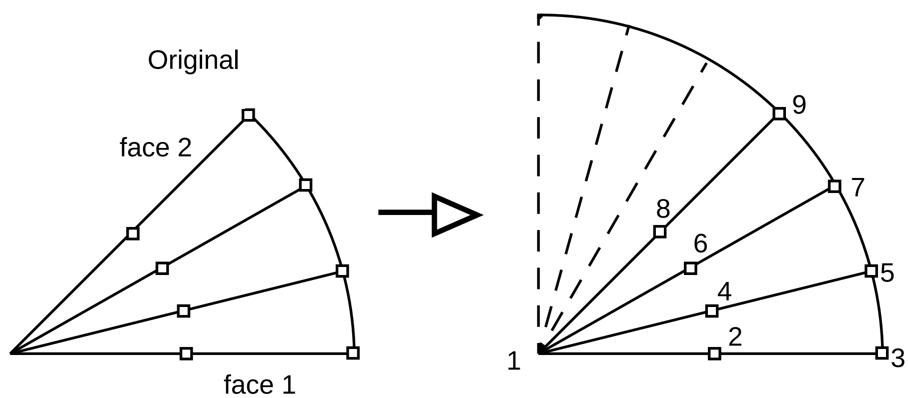

Figure 2-25
Model Duplication by rotational symmetry

The periodic match file for a rotation of this model about point 1 would be:

```
rotate
45.0
3
1 1
2 8
3 9
```

Thus, face 1 of this model is made up of nodes 1, 2, and 3 and face 2 of this model is made up of nodes 1, 8, and 9

so there are 3 node pairs to define

with node 1 corresponding to node 1 after a copy is rotated,
node 2 corresponding to node 8, and
node 3 corresponding to node 9

Default Part Colors File Format

This file defines default Constant Colors that are assigned (and cycled through) by EnSight when parts are built. If EnSight does not find a definition file it uses an internal default list. If, however, EnSight does find a file (the file must be called `ensight.part.colors.default` and be located in the `.ensight6` directory of the user's home directory or be located in `$ENSIGHT6_HOME/site_preferences`) at start-up, EnSight will read your colors as the default Constant Colors.

The format of the `ensight.part.colors.default` file is as follows:

- Line 1: "Version 6.0"
- Line 2: One integer, the number of default part colors
- Line 3 on: three floats (each ranging between 0. and 1.), the (red, green, blue) color triplet of each color, each listed on separate lines.

An example default file can be found in:

`$ENSIGHT6_HOME/site_preferences/ensight.part.colors.default`

on your client system.

The following is an example default part colors file with 6 colors (blue, cyan, green, yellow, red, and magenta):

```
Version 6.0
6
0. 0. 1.
0. 1. 1.
0. 1. 0.
1. 1. 0.
1. 0. 0.
1. 0. 1.
```

Default False Color Map File Format

This file defines the default false-color map color range that is assigned by EnSight to each palette when variables are activated. If EnSight does not find a definition file, it uses an internal default list. If, however, EnSight does find a file (the file must be called `ensight.false_color.default` and be located in the `.ensight6` directory of the user's home directory or be located in `$ENSIGHT6_HOME/site_preferences`) at start-up, EnSight will read your colors as the default palette colors.

The format of the `ensight.part.colors.default` file is as follows:

- Line 1: "Version 6.0"
- Line 2: One integer, the number default false color map colors
- Line 3 on: three floats (each ranging between 0. and 1.), the (red, green, blue) color triplet of each color, each listed on separate lines.

An example default file can be found in:

`$ENSIGHT6_HOME/site_preferences/ensight.false_color.default`

on your client system.

The following is an example default false color map file with 5 colors; blue, cyan, green, yellow, and red:

```
Version 6.0
5
0. 0. 1.
0. 1. 1.
0. 1. 0.
1. 1. 0.
1. 0. 0.
```

MPEG Parameters File

This file sets the parameters used by the MPEG Encoder. MPEG is a lossy video compression standard. As such, there are trade-offs to be made regarding the degree of compression vs. image quality. The MPEG Encoder utilizes a parameters file to set numerous options that affect quality, compression, and other attributes. Three sample parameter files can be found in:

`$ENSIGHT6_HOME/site_preferences/`

These files roughly correspond to:

high quality/low compression (`cei_mpeg_hi_q.param`),
medium quality/medium compression (`cei_mpeg_med_q.param`),
and low quality/high compression (`cei_mpeg_lo_q.param`).

The format of the parameters file is documented in the PostScript document:

`$ENSIGHT6_HOME/doc/mpeg/mpeg_encode_doc.ps`

The Encoder will read the parameters from `~/.ensight6/cei_mpeg.param` if it exists, otherwise it will use `$ENSIGHT6_HOME/site_preferences/cei_mpeg.param` which is a link to `cei_mpeg_hi_q.param`; thus high quality/low compression is the default.

Note: *This is opposite of the EnSight 6.1 default.*

Please see the file:

`$ENSIGHT6_HOME/doc/mpeg/README.mpeg`

for further information.